Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждениевысшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, математики и естественно-научных дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
___ Е. Н. Скуратенко
подпись
«____» _____ 2022 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 Прикладная информатика

Разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

Руководитель _____ доцент, канд. физ.-мат. наук М. А. Буреева
подпись, дата

Выпускник _____ А. Д. Сагояков
подпись, дата

Консультанты
по разделам:

Экономический _____ Е.Н. Скуратенко
подпись, дата

Нормоконтролер _____ В. И. Кокова
подпись, дата

Абакан 2022

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждениевысшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, математики и естественно-научных дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
___ Е. Н. Скуратенко
подпись
«____» _____ 2022 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Абакан 2022

Студенту Сагоякову Александру Дмитриевичу

Группа ХБ 18-03

Направление 09.03.03 Прикладная информатика

Тема выпускной квалификационной работы:Разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

Утверждена приказом по институту № 208 от 14.04.2022 г.

Руководитель ВКР: М. А. Буреева, доцент, канд. физ.-мат. наук, ХТИ – филиал СФУ

Исходные данные для ВКР: заказ ГБУЗ РХ «РМИАЦ».

Перечень разделов ВКР:

1. Анализ разработки «Паспорта цифровизации ГБУЗ РХ «РМИАЦ».
2. Разработка информационной системы.
3. Расчёт затрат реализации проекта «Паспорт цифровизации ГБУЗ РХ «РМИАЦ».

Перечень графического материала: нет

Руководитель ВКР _____ М. А. Буреева
подпись

Задание принял к исполнению_____ А. Д. Сагояков
подпись

«14» апреля 2022 г.

РЕФЕРАТ

Выпускная квалификационная работа (ВКР) по теме «Разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ» содержит 73 страницы текстового документа, 20 таблиц, 45 иллюстраций, 8 формул, 14 использованных источников, 1 приложение.

ПАСПОРТ ЦИФРОВИЗАЦИИ, ИНФОРМАЦИОННАЯ СИСТЕМА, PYTHON, БАЗА ДАННЫХ, ЗАТРАТЫ, РИСКИ, ЭКОНОМИЧЕСКАЯ ЭФФЕКТИВНОСТЬ.

Объект выпускной квалификационной работы: деятельность сотрудников ГБУЗ РХ «РМИАЦ» по учету технической оснащенности медицинских организаций.

Предмет ВКР: процесс автоматизации учета инфраструктуры медицинских организаций по Республике Хакасия.

Цель ВКР: разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ» для учета инфраструктуры медицинских организаций Республики Хакасия.

В первом разделе ВКР описывается основная деятельность ГБУЗ РХ «РМИАЦ». Проанализированы средства для создания информационной системы «паспорт цифровизации». Выбраны средства разработки информационной системы «паспорт цифровизации».

Во втором разделе был описан процесс установки необходимого программного обеспечения. Поэтапно описан процесс разработки информационной системы, а также протестирован итоговый вариант.

В третьем разделе выполнен расчет затрат реализации проекта по методу ТСО, проанализированы риски, которые могут возникнуть в процессе разработки и эксплуатации, представлены возможные мероприятия по их решению.

В результате была разработана информационная система «паспорт цифровизации ГБУЗ РХ «РМИАЦ».

SUMMARY

The theme of the graduation thesis is "Development of the information system "Digitalization Passport of the State Budgetary Health Institution of the Republic of Khakassia «Republican Medical Information and Analytical Center»". It consists of 73 pages, 45 figures, 20 tables, 8 formulae, 14 reference items, 1 appendix.

PASSPORT OF DIGITALIZATION, INFORMATION SYSTEM, PYTHON, DATABASE, COSTS, RISKS, ECONOMIC EFFICIENCY.

The object of the thesis: the activities of employees of GBUZ RH "RMIATS" on accounting for the technical equipment of medical organizations.

Thesubjectofthethesis:the process of automating the accounting of the infrastructure of medical organizations in the Republic of Khakassia.

The purpose of the thesis is to develop of the information system "Digitalization Passport of the GBUZ RH "RMIATS" to account for the infrastructure of medical organizations of the Republic of Khakassia.

The second section described the main activities of the RMIAC. The means for creating an information system "digitalization passport" are analyzed. The means of developing the information system "digitalization passport" have been selected.

The second section described how to install the required software. The process of developing an information system is described step by step, and the final version is tested.

In the third section, the cost of project implementation is calculated using the total cost of ownership method, analyzes the risks that may arise during development and operation, and presents possible measures to address them.

As a result, the information system “Digitalization Passport of the GBUZ RH "RMIATS" was developed.

English language supervisor _____ N.V. Chezybaeva

Signature, date

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Единая государственная информационная система в сфере здравоохранения (ЕГИСЗ)включает в себя информационное обеспечение в сфере здравоохранения посредством создания, развития и эксплуатации на всех уровнях, в том числе и на региональном, информационных систем медицинских организаций[11].

Одной из подсистем ЕГИСЗ является Федеральный реестр медицинских организаций. Он предназначен для учета сведений о медицинских организациях государственной, муниципальной и частной систем здравоохранения, их структурных подразделениях с указанием профилей их медицинской деятельности, местонахождения, сведений об их оснащении, использования медицинских изделий [11].

В Республике Хакасия за создание подобной информационной системы отвечает ГБУЗ РХ«Республиканский медицинский информационно-аналитический центр» (ГБУЗ РХ «РМИАЦ») [13].

Объект выпускной квалификационной работы: деятельность сотрудников ГБУЗ РХ «РМИАЦ» по учету технической оснащенности медицинских организаций.

Предмет ВКР: процесс автоматизации учета инфраструктуры медицинских организаций по Республике Хакасия.

Цель ВКР: разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ» для учета инфраструктуры медицинских организаций Республики Хакасия.

Для достижения поставленной цели были решены следующие задачи:

– проведен анализ деятельности ГБУЗ РХ «РМИАЦ»;

– выделены показатели цифровизации медицинских организаций Республики Хакасия;

– разработана информационно-логическая модель базы данных для хранения сведений о медицинских организациях Республики Хакасия;

– проанализированы средства и осуществлен выбор для создания информационной системы «Паспорт цифровизации»;

– выполнена установка необходимого программного обеспечения;

– выполнена настройка и заполнение базы данных;

– настроен вывод информации из базы данных на экран по запросу пользователя;

– реализован отчет о заполненности таблиц;

– настроена фильтрация данных в системе.

В первом разделе ВКР описывается основная деятельность ГБУЗ РХ «РМИАЦ». Проанализированы средства для создания информационной системы «Паспорт цифровизации». Выбраны средства разработки информационной системы «Паспорт цифровизации».

Во втором разделе был описан процесс установки необходимого программного обеспечения. Поэтапно описан процесс разработки информационной системы, а также протестирован итоговый вариант.

В третьем разделе выполнен расчет затрат реализации проекта по методу ТСО, проанализированы риски, которые могут возникнуть в процессе разработки, представлены возможные мероприятия по их решению.

1 Анализ разработки «Паспорта цифровизации ГБУЗ РХ «РМИАЦ»

1.1 Анализ деятельности ГБУЗ РХ «РМИАЦ» в рамках создания «Паспорта цифровизации»

В рамках государственной программы «Цифровая трансформация в сфере здравоохранения» планируется создать механизмы взаимодействия медицинских организаций на основе единой государственной информационной системы в сфере здравоохранения, целью которых является обеспечение цифровой трансформации и повышение эффективности функционирования отрасли на всех уровнях и создания условий для использования гражданами электронных услуг и сервисов в сфере здравоохранения [12].

В рамках федерального проекта предстоит решение задач по трансформации процессов организации системы здравоохранения за счет автоматизированного информационного сопровождения, а также мониторинга и анализа использования ресурсов здравоохранения и оказания медицинской помощи гражданам.

Одной из организаций, попадающих под программу, является ГБУЗ РХ «РМИАЦ».

ГБУЗ РХ «РМИАЦ» – это республиканский медицинский информационно-аналитический центр, который занимается сбором, анализом и обработкой медицинских данных о деятельности медицинских организаций в республике Хакасия. Помимо этого, РМИАЦ собирает данные о состоянии здоровья населения, управляет системой медицинского статистического учета и отчетности в организациях и учреждениях здравоохранения Республики Хакасия. Исходя из выше сказанного можно выделить следующие направления работы РМИАЦ[10]:

- внедрение и сопровождение медицинских информационных систем;
- обеспечение информационной поддержки информационных систем;
- настройка и обслуживание медицинских информационных систем;

− координация работ с разработчиками и с обладателями медицинских информационных систем;

− разработка инструкций, регламентов программного обеспечения;

− разработка (совместно с соответствующими подразделениями) мероприятий по совершенствованию форм и методов работы с медицинскими информационными системами;

− организация своевременного рассмотрения и исполнения заявок на выполнение работ, связанных с функционированием программного обеспечения;

− обеспечение технической поддержки электронной записи на прием к врачу;

− обеспечение технической поддержки для медицинских организаций Республики Хакасия по работе с информационными системами.

Для выполнения поставленных задач РМИАЦ нужны эффективные способы сбора и управления информацией о медицинских организациях. Для достижения этой цели требуются системы, управляющие данными не только внутри одной медицинской организации, но и система, способная хранить и взаимодействовать в одном вместе с данными обо всех медицинских организациях.

Для обеспечения большого круга задач, стоящих перед РМИАЦ, на предприятии имеется несколько отделов: отдел программного обеспечения, отдел медицинской статистики и мониторинга, администрация руководства, администрация при работе с информацией и отдел управления инфраструктурой.

Наглядно это отражается в структурной схеме предприятия, представленной на рисунке 1.

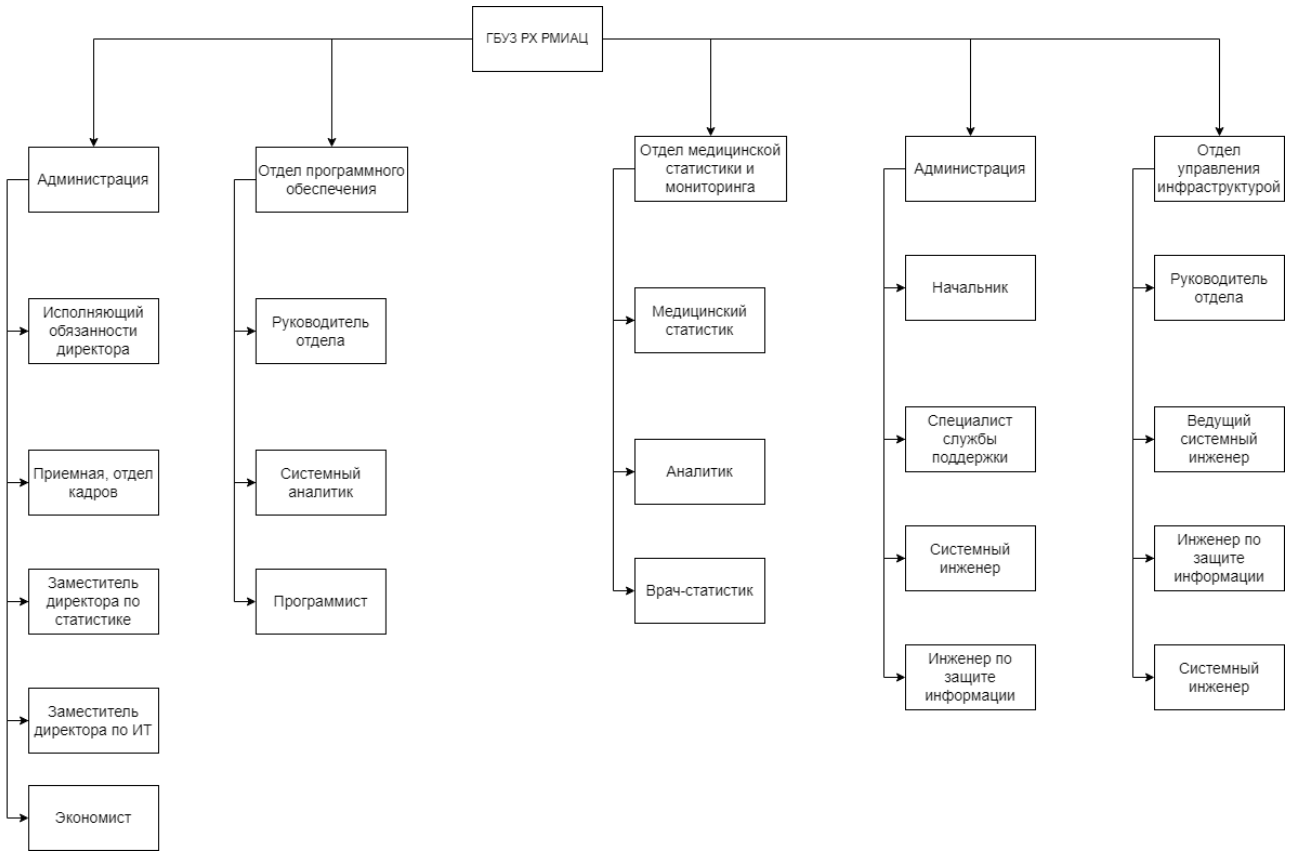

Рисунок 1 – Структурная схема ГБУЗ РХ «РМИАЦ»

1.2 Информационная система «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

Отдел управления инфраструктурой занимается учетом медицинских организацийи их инфраструктурного обеспечения. В перечень необходимой информации входят:

– общая информация об организации (адрес, уровень медицинской организации)

– информация об используемых в медицинской организации информационных системах;

– территориально-выделенные структурные подразделения (ТВСП);

– информация о защищенности локальной сети и сети интернет;

– оснащенность оборудованием, а именно: технические характеристики компьютеров, периферия, информация об используемом программном обеспечении.

Для создания системы, способной взаимодействовать с вышеуказанной информацией, требуется спроектировать и создать соответствующую базу данных.

В базе данных «Паспорт цифровизации» определены 13 сущностей:

– «Основные сведения о медицинских организациях»;
– «Информационные системы»;
– «ТВСП»;
– «Диагностическое оборудование»;
– «Защищенная сеть передачи данных»;
– «Информация о доступе в Интернет»;
– «Локальная вычислительная сеть»;
– «Оборудование АРМ»;
– «Системный блок»;
– «Монитор»;
– «Компьютерная мышь»;
– «Клавиатура»;
– «Программное обеспечение».

У каждой медицинской организации может быть несколько информационных систем и ТВСП. Кроме того, в каждом ТВСП несколько видов диагностического оборудования. Информация о защищенной сети передачи данных, доступе в Интернет, локальной вычислительной сети и оборудовании АРМ составляет сетевую инфраструктуру выделенного ТВСП. В свою очередь оборудование АРМ подразделяется на системный блок, монитор, компьютерную мышь, клавиатуру и программное обеспечение.

Сбор сведений о медицинской организации начинается с записи общей информации в таблице «Медицинская организация», куда записываются

название, адрес, идентификатор объекта или OID, уровень организации и idадминистратора.

Таким образом, атрибуты сущности «Основные сведения о медицинских организациях»:

1. Наименование.
2. Юридический адрес.
3. Идентификатор медицинской организации в системе РМИАЦ (OIDMO).
4. Уровень медицинской организации.
5. Id администратора медицинской организации.

Возможным ключом, полностью определяющим объект отношения «Медицинская организация» может служить НаименованиеилиOID МО. Однако возможный ключ в обоих случаях имеетслишком большую размерность для использования его в качестве первичногоключа, поэтому был введен дополнительный атрибут Номер_МО и выбран в качестве первичного ключа. Все значения этого атрибута должны быть уникальными.

Определим имена полей и их типы таблицы «municipal_institutions»:

1. Номер_МО: имя поля mo_id, тип числовой.
2. Наименование: имя поля name, тип символьный.
3. Юридический адрес: имя поля address, тип символьный.
4. OIDMO: имя поля oid_municipal, тип символьный.
5. Уровень медицинской организации: имя поля level_municipal, тип символьный.

Аналогичным образом были определены остальные таблицы базы данных.

Сущность «Информационные системы» содержит информацию об информационных системах, использующихся в данной организации: наименование системы, её тип, разработчик, наличие или отсутствие модели угроз, информация об аттестате соответствия Федеральной службы по техническому и экспортному контролю (ФСТЭК), является ли система

геоинформационной (ГИС), входит ли система в критическую информационную инфраструктуру (КИИ), и если да, то какая у нее категория.

В таблице 1 приводится соответствие между атрибутами сущности «Информационные системы» и полями таблицы InfSystems. Первичным ключом является поле infsys_id.

Таблица 1 – Соответствие между атрибутами сущности «Информационные системы» и полями таблицы InfSystems

Атрибут сущности «Информационные системы»	**Имя поля таблицы InfSystems**	**Тип поля таблицы InfSystems**
Идентификатор информационной системы	infsys_id	числовой
Название ИС	name	символьный
Тип ИС	type	символьный
Разработчик ИС	developer	символьный
Наличие модели угроз	treat_model_presence	boolean
Наличие аттестата соответствия ФТСЭК	conformity_certificate	boolean
ГИС	gis	boolean
Объект КИИ	object_cii	boolean
Категория КИИ	cii_category	символьный

В таблице «ТВСП» записывается информация о территориально выделенных структурных подразделениях (ТВСП), существующих в медицинской организации. Помимо названия, типа и адреса, в ТВСП учитывается наличие или отсутствие какого-либо отделения, а также записывается общая информация о здании: количество этажей, состояние и планы на реорганизацию. Первичным ключом является поле tvsp_id (таблица 2).

Таблица 2 – Соответствие между атрибутами сущности «ТВСП» и полями таблицы tvsp

Атрибут сущности «ТВСП»	**Имя поля таблицы tvsp**	**Тип поля таблицыtvsp**
Идентификатор ТВСП	tvsp_id	числовой
Название ТВСП	name_tvsp	символьный
Тип ТВСП	type_tvsp	символьный
Адрес ТВСП	adress_tvsp	символьный
OID ТВСП	oid_tvsp	символьный
Координаты ТВСП	coordinates_tvsp	символьный
Количество этажей ТВСП	gis	символьный
Форма собственности	object_cii	символьный
Состояение здания	cii_category	символьный
Количество этажей ТВСП	floors_tvsp	символьный
Форма собственности	own_form	символьный
Состояение здания	building_status	символьный
Планируется реорганизация	planned_reorganization	символьный
Амбулаторная помощь	medical_care	boolean
Оказание ПМСП	Provision_of_PHC	boolean
Наличие регистратуры	registration_temporary_incapacity	boolean
Стационарная помощь	sanitary_protection_zone	boolean
КДЛ	kdl	boolean
СМП	smp	boolean
НМП	nmp	boolean
Инструментальная диагностика	instrumental_diagnostic	boolean
ЛЛО	llo	boolean
Направление на МСЭ	referral_to_itu	boolean
Оформление листов временной нетрудоспособности	disability_list	boolean
ССЗ	ssz	boolean
Онкология	cancer	boolean

Далее, в зависимости от выбранного подразделения определённой медицинской организации, записывается информация о сетевой инфраструктуре и диагностическом оборудовании.

В таблице «Диагностическое оборудование» записывается наименование и тип оборудования, год выпуска и место размещения, использовалось ли оборудование и есть ли человек, отвечающий за него.Первичным ключом является поле diagnostic_id (таблица 3).

Таблица 3 – Соответствие между атрибутами сущности «Диагностическое оборудование» и полями таблицы DiagnosticEqip

Атрибут сущности «Диагностическое оборудование»	**Имя поля таблицы DiagnosticEqip**	**Тип поля таблицыDiagnosticEqip**
Идентификатор оборудования	diagnostic_id	числовой
Название ИС	name	символьный
Тип ИС	type	символьный
Год выпуска	release_year	дата
Место размещение оборудования	location	символьный
Наличие подключения к ЛВС	lvs_connect_capasity	boolean
Количество исследований в год	research_year_number	символьный
Средний объем одного исследования (МБ)	average_size_research	символьный
Признак активации DICOM 3.0	sign_activation_discom	boolean
Наличие АРМ диагноста	existence_armd_discom	boolean

Таблица «Сетевая инфраструктура» является связующей между таблицей «ТВСП» и таблицами, содержащими данные о сетевых подключениях и оборудовании АРМ.

В таблице «Защищенная сеть передачи данных» содержится информация о защищенности внутренней сети: является ли сеть защищенной, какая технология используется. VipNet– программный комплекс, который

предназначен для защиты пользователей от внешних угроз. Первичным ключом является security_network_id (таблица 4).

Таблица 4 – Соответствие между атрибутами сущности «Защищенная сеть передачи данных» и полями таблицы security_network

Атрибут сущности «Защищенная сеть передачи данных»	**Имяполятаблицыsecurity_network**	**Типполятаблицыsecurity_network**
Идентификатор сети	security_network_id	числовой
Статус подключения к ЗСПД	status_zcpd_connect	boolean
Технология подключения к ЗСПД	vipnet_coordinator	boolean
ViPNet Coordinator	vipnet_hw1000	boolean
ViPNet HW1000	vipnet_hw100	boolean
iPNet Client	vipnet_client	boolean

Таблица «Информация о доступе к сети Интернет» содержит информацию о внешнем подключении, а именно: присутствует ли подключение к сети Интернет, какая технология используется, скорость и стоимость услуг, а также наименование поставщика, оказывающего услугу подключения. Первичным ключом является network_connect_id (таблица 5).

Таблица 5 – Соответствие между атрибутами сущности «Информация о доступе к сети Интернет» и полями таблицы network_connect

Атрибут сущности	**Имя поля таблицы**	**ТипполятаблицыDiagnosticEqip**

«Информация о доступе к сети Интернет»	**network_connect**	
Идентификатор доступа к сети	network_connect_id	числовой
Статус подключения к интернет	status_interent_connect	символьный
Технология подключения к интернет	tech_internet_connect	символьный
Пропускная способность канала	channel_capacity	символьный

Окончание таблицы 5

Пропускная способность по тарифу	tariff_capacity	символьный
Поставщик услуг	service_provider	символьный
Стоимость услуг в месяц	cost_in_month	символьный
Участие в ФП	participation_in_fp	символьный

Таблица «Локальная вычислительная сеть» хранит общую информацию о локальной сети внутри структурного подразделения. Первичным ключом является local_network_id (таблица 6).

Таблица 6 – Соответствие между атрибутами сущности «Локальная вычислительная сеть» и полями таблицы local_network

Атрибут сущности «Локальная вычислительная сеть»	**Имя поля таблицы local_network**	**Типполятаблицыlocal_network**
Идентификатор ЛВС	local_network_id	числовой
Количество портов ЛВС	number_port_lvs	символьный
Сетевое оборудование	network_equipment	символьный
IP-адресация	ip_connect	символьный
Площадь помещения	place	символьный

Место размещения	server_room_compliance	символьный
Соответствие требования к серверным помещениям	standart_rule	символьный
Наличие схемы ЛВС	LVS	boolean

Таблица «АРМ» является связующей и хранит информацию об оборудовании и кабинете, в котором оно хранится.Первичным ключом является arm_id (таблица 7).

Таблица 7 – Соответствие между атрибутами сущности «АРМ» и полями таблицы Arm

Атрибут сущности «АРМ»	**Имя поля таблицы Arm**	**ТиполятаблицыArm**
Идентификатор АРМ	arm_id	числовой
Номер кабинета	number_room	символьный

Одной из связанных таблиц является таблица «Системный блок», где расположена информация о дате приобретения, инвентарном номере, модели процессора, информации об оперативной памяти и системе хранения, а также содержит информацию об операционной системе.Первичным ключом является sb_id (таблица 8).

Таблица 8 – Соответствие между атрибутами сущности «Системный блок» и полями таблицы SystemBlock

Атрибут сущности «Информация о доступе к сети Интернет»	**Имя поля таблицы SystemBlock**	**ТиполятаблицыSystemBlock**
Идентификатор системного блока	sb_id	числовой
Дата покупки	purchase_year	символьный

Инвентарный номер	inventory_number	символьный
Модель процессора	processor_model	символьный
Модель ОЗУ	model_ram	символьный
Объем ОЗУ	ram_size	символьный
Тип системы хранения	storage_system_type	символьный
Объем хранения	storage_system_size	символьный
Операционная система	operating_system	символьный

Таблица «Монитор» содержит информацию о модели, диагонали, инвентарном номере и дате приобретения монитора.Первичным ключом является monitor_id (таблица 9).

Таблица 9 – Соответствие между атрибутами сущности «Монитор» и полями таблицы Monitor

Атрибут сущности «Монитор»	**Имя поля таблицы Monitor**	**ТиполяталицыMonitor**
Идентификатор монитора	monitor_id	числовой
Модель	model	символьный
Диагональ	diagonal	символьный
Инвентарный номер	inventory_number	символьный
Год покупки	purchase_year	символьный

Таблицы «Компьютерная мышь» и «Клавиатура» содержат информацию о модели, инвентарном номере и дате приобретения соответствующего оборудования.Первичным ключом для таблицы «Компьютерная мышь» является mouse_id (таблица 10), для таблицы «Клавиатура» – keyboard_id (таблица 11).

Таблица 10 – Соответствие между атрибутами сущности «Компьютерная мышь» и полями таблицы Mouse

Атрибут сущности «Компьютерная мышь»	**Имя поля таблицы Mouse**	**ТиполяталицыMouse**

Идентификатор мыши	mouse_id	числовой
Модель	model	символьный
Инвентарный номер	inventory_number	символьный
Год покупки	purchase_year	символьный

Таблица 11 – Соответствие между атрибутами сущности «Клавиатура» и полями таблицы Keyboard

Атрибут сущности «Компьютерная мышь»	**Имя поля таблицы Keyboard**	**ТиполяталицыKeyboard**
Идентификатор клавиатуры	keyboard_id	числовой
Модель	model	символьный

Окончание таблицы 11

Инвентарный номер	inventory_number	символьный
Год покупки	purchase_year	символьный

В таблице «Программное обеспечение» содержится информация об установленном программном обеспечении, включающая в себя название, наименование разработчика, стоимость, версию, наличие в реестре отечественного ПО. Первичным ключом является software_id (таблица 12).

Таблица 12 – Соответствие между атрибутами сущности «Программное обеспечение» и полями таблицы Software

Атрибут сущности «Программное обеспечение»	**Имя поля таблицы Software**	**ТиполяталицыSoftware**
Идентификатор программного обеспечения	software_id	числовой
Название	name	символьный
Разработчик	developer	символьный
Наличие в реестре отечественного ПО	home_soft	boolean
Версия	verison	символьный

Цена	price	символьный
Год покупки	purchase_year	символьный

Созданная информационно-логическая модель, отражающая связь таблиц между собой в базе данных, представлена на рисунке 2.

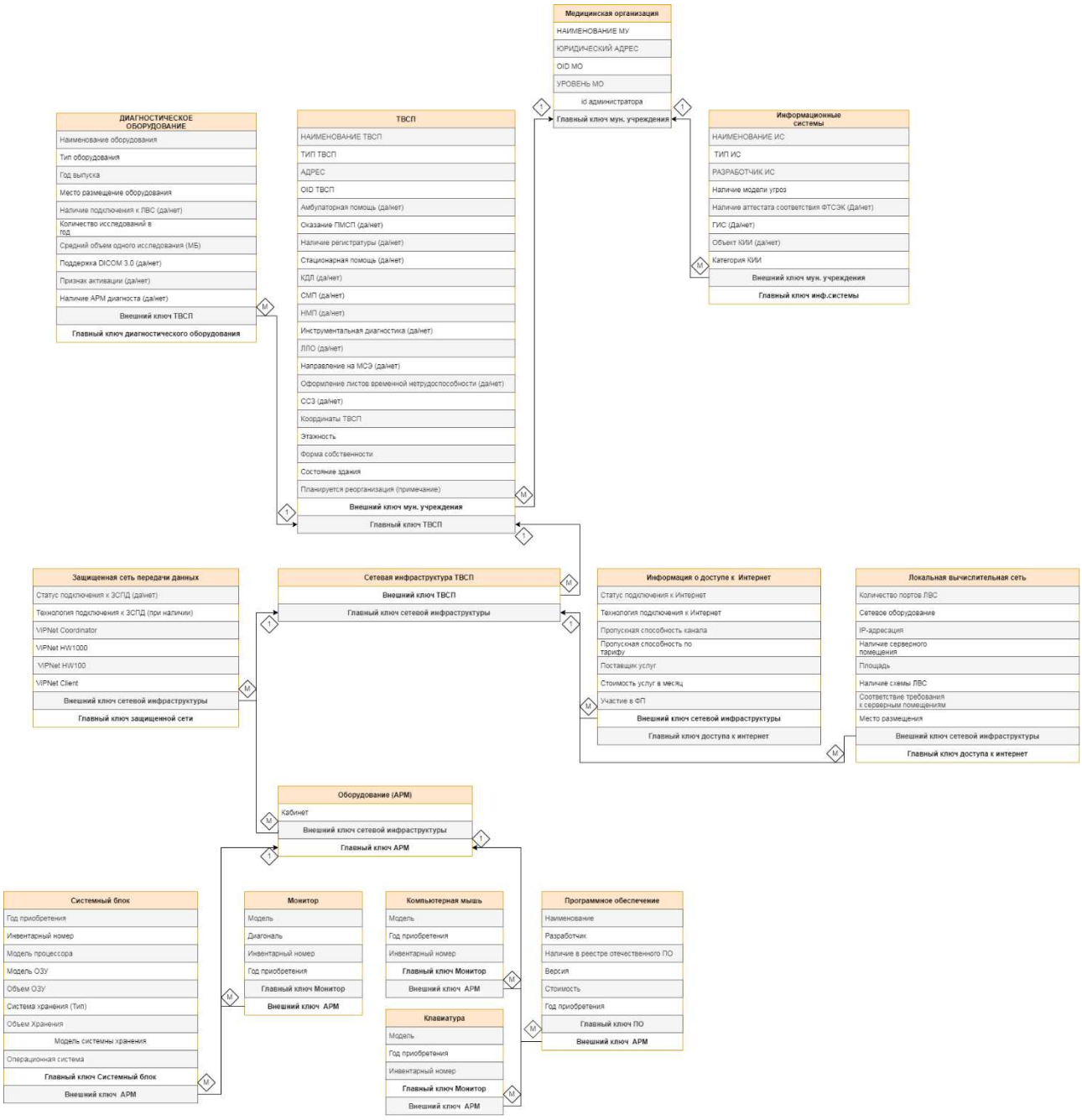

Рисунок 2 – Информационно-логическая модель базы данных

Помимо заполнения базы данных, требуется постоянная проверка актуальности данных, а также проверка степени заполненности таблиц, формирование и вывод отчетов об инфраструктурном обеспечении медицинских организаций.

1.3 Обзор программных средств для разработки паспорта цифровизации

Перед созданием базы данных были рассмотрены следующие средства управления базами данных: MySQL[3], SQLite [8], PostgreSQL[5].

Mysql – является одной из самой популярных СУБД из серверных баз данных. Достаточно проста в изучении, благодаря ее популярности можно найти множество документации в сети интернет. Однако, несмотря на это, у MySQL есть несколько минусов, таких как: проблемы с совместимостью с другими СУБД; некоторые версии MySQL могут не содержать нужные функции.

PostgreSQL – является одной из самых продвинутых СУБД, которая постоянно развивается, также обладает объектно-ориентированным функционалом. Однако в простых операциях чтения данная СУБД уступает другим аналогам, несмотря на продвинутый функционал PostgreSQL не очень популярная по сравнению с остальными, а значит будет труднее искать информацию по ней.

Для создания базы данных была выбрана система управления базами данных SQLite. Выбор основывался на нескольких критериях:

– За счет своей особенности в архитектуре SQLite работает достаточно быстро, что позволяет оперативно проверять сделанные изменения в базе данных.

– Созданная база хранится в одном файле, а значит ее перемещение не составит практически никакого затруднения.Т.к. система разрабатывалась на

локальном сервере, возможность быстрого переноса данных практически необходима.

– Перед использованием СУБД не нужна сложная настройка или длительная установка.

– СУБД является кроссплатформенной и подходит для различных систем.

– Система независима от сторонних фреймворков, программного обеспечения или библиотек, для работы не требуются дополнительные компоненты.

Помимо создания базы данных требуется создать информационную систему, взаимодействующую с этой базой.Информационная система должна быть в виде веб-приложения, чтобы доступ к ней был осуществим из любой медицинской организации.

Так как база данных содержит достаточно большой объем информации, эта информация постоянно обновляется и обрабатывается, то для работы с ней понадобится язык программирования, подходящий для анализа больших объемов данных. В настоящее время эта задача легче всего решается с помощью языкаPython [7]. Кроме того,одной изосновных сфер применения этого языка является веб-разработка.В отличие от языка PHP [4], который тоже используется в веб-разработке, в Pythonесть улучшенная структура и легкодоступные инструменты для отладки языка. Множество доступных библиотек позволяет не только сократить программный код, но и облегчить обработку данных.

Однако, помимо языка для веб-разработки, понадобится знание фреймворков, программных платформ, которые определят структуру системы. Наиболее популярными являются фреймворки Django [1]и Flask [2]. Фреймворк Djangoпредоставляет собственное объектно-реляционное отображение или ORMи использует модели данных, которые позволяют разработчикам связывать таблицы в базе данных с классами в языке программирования. Это позволит работать с моделями,используяссылки на базу данных. У Flasktакой

модели нет, так как ее основное применение – это создание одностраничного приложения или личных блогов. Исходя из этого, был выбран фреймворк Django. Разработка приложения велась с интегрированной среде разработки PyCharm[6].

1.4 Выводы по разделу «Анализ разработки «Паспорта цифровизации ГБУЗ РХ «РМИАЦ»

Таким образом, анализ деятельности ГБУЗ РХ «РМИАЦ» позволил определить основные показатели цифровизации медицинских организаций Республики Хакасия. На основании этого была построена информационно-логическая модель базы данных об их инфраструктурном обеспечении: определены таблицы, связи между ними. Проведен анализ программного обеспечения, необходимого для разработки информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ». Осуществлен выбор средств разработки.

2 Разработка информационной системы

2.1 Установка необходимого программного обеспечения

Для установки Djangoпотребуется пакетный менеджер pip, который позволит загружать пакеты и управлять ими. Для установки нужно ввести в панели администратора команду: python–m ensurepip–upgrade или, если pipбыл установлен ранее, команду: pipinstall–upgradepip.

Далее нужно установить виртуальную среду. Виртуальная среда, илиvenv,позволит запускать разные приложения независимо друг от друга. Установить его можно введя команду: python–mvenvhello, гдehello –это выбранное название виртуальной среды. Затем виртуальную среду нужно

активировать, пройдя по пути hello/scripts/.В папке scriptsбудет файл activate.bat.Активировав его через командную строку, запустится виртуальная среда.

После активации виртуальной среды для установки Djangotребуется ввести следующую команду: python–mpipinstallDjango.

После запуска проекта для создания базы данных нужно перейти к файлу models.py, который расположен внутри папки с названием проекта. В этом файле описываются таблицы базы данных и их настройки, такие как название столбцов, тип водимых данных, длина ввода и значение по умолчанию. На рисунке 3представлен программный код для создания таблицы «municipal_institutions»(«Медицинская организация»).

```
class municipal_institutions(models.Model):
    municipal_id = models.IntegerField(u'Код учреждения', primary_key=True)

    name = models.CharField(u'Название учреждения',max_length=255,default = 'Заполните')
    address = models.CharField(u'Адрес учреждения',max_length=255,default = 'Заполните')
    oid_municipal = models.CharField(u'OID учреждения', max_length=255,default = 'Заполните')
    level_municipal = models.CharField(u'Уровень',max_length=255,default = 'Заполните')
```

Рисунок3 – Созданиетаблицы«municipal_institutions»

Поле «municipal_id» являетсяпервичнымключомдлятаблицы «municipal_institutions», типданныхдлявводанаписанспомощьюкоманды «models.IntegerField», гдеInteger– типвводимыхданных,вданномслучае– целочисленныйтип. В кавычках указано название столбца, которое будет отображено при вводе информации в базу данных. Параметр «max_length» соответствует максимально допустимому количеству вводимых символов.

Следующей была создана таблица «InfSystems» («Информационные системы»).Первичным ключом является поле «infsys_id», а поле «municipal_institutions» – внешним ключом, который связывает эту таблицу с таблицей «municipal_institutions». На рисунке 4 представлен фрагмент кода для создания таблицы.

```python
class InfSystems(models.Model):
    infsys_id = models.IntegerField(
        u'ID информационной системы мун.учреждения',
        db_column='infsys_id',
        primary_key=True
    )
    name = models.CharField(u'Название ИС',max_length=255,default = 'Заполните')
    type = models.CharField(u'Тип ИС',max_length=255,default = 'Заполните')
    developer = models.CharField(u'Разработчик ИС',max_length=255,default = 'Заполните')
    treat_model_presence = models.CharField(u'Наличие модели угроз',max_length=255,default = 'Заполните')
    conformity_certificate = models.CharField(u'Наличие аттестата соответствия ФТСЭК',max_length=255,default = 'Заполните')
    gis = models.BooleanField(u'ГИС?',default=False)
    object_cii = models.BooleanField(u'Объект КИИ?',default=False)
    cii_category = models.CharField(u'Категория КИИ',max_length=255,default = 'Заполните')

    municipal_institutions = models.ForeignKey('municipal_institutions', models.PROTECT,verbose_name=u"id Мун.учреждения", null = True)
```

Рисунок 4 – Создание таблицы «InfSystems»

Далее была создана таблица «tvsp» («ТВСП») с первичным ключом «tvsp_id» и внешним ключом «municipal_institutions».Некоторые поля в этой таблице, такие как «kdl»или «smp» имеют тип Boolean,так как в этих полях указывается наличие или отсутствие каких-либо элементовв ТВСП, а потому здесь только два варианта вводимых данных. На рисунке 5 представлено создание таблицы «tvsp».

```python
class tvsp(models.Model):
    tvsp_id = models.IntegerField(
        db_column='tvsp_id',
        primary_key=True
    )
    name_tvsp = models.CharField(u'Название ТВСП', max_length=255,default = 'Заполните')
    type_tvsp = models.CharField(u'Тип ТВСП',max_length=255,default = 'Заполните')
    adress_tvsp= models.CharField(u'Адрес ТВСП',max_length=255,default = 'Заполните')
    oid_tvsp = models.CharField(u'OID ТВСП',max_length=255,default = 'Заполните')
    coordinates_tvsp = models.CharField(u'Координаты ТВСП',max_length=255,default = 'Заполните')
    floors_tvsp = models.CharField(u'Количество этажей ТВСП',max_length=255,default = 'Заполните')
    own_form = models.CharField(u'Форма собственности',max_length=255,default = 'Заполните')
    building_status = models.CharField(u'Состояние здания',max_length=255,default = 'Заполните')
    planned_reorganization = models.CharField(u'Планируется реорганизация?',max_length=255,default = 'Заполните')
    medical_care = models.BooleanField(u'Амбулаторная помощь',default=False)
    Provision_of_PHC = models.BooleanField(u'Оказание ПМСП',default=False)
    registration_temporary_incapacity = models.BooleanField(u'Наличие регистратуры',default=False)
    sanitary_protection_zone = models.BooleanField(u'Стационарная помощь',default=False)
    kdl = models.BooleanField(u'КДЛ',default=False)
    smp = models.BooleanField(u'СМП',default=False)
    nmp = models.BooleanField(u'НМП',default=False)
    instrumental_diagnostic = models.BooleanField(u'Инструментальная диагностика',default=False)
    llo = models.BooleanField(u'ЛЛО',default=False)
    referral_to_itu = models.BooleanField(u'Направление на МСЭ',default=False)
    disability_list = models.BooleanField(u'Оформление листов временной нетрудоспособности',default=False)
    ssz = models.BooleanField(u'ССЗ',default=False)
    cancer = models.BooleanField(u'Онкология',default=False)

    municipal_institutions = models.ForeignKey('municipal_institutions', models.PROTECT,verbose_name=u"id Мун.учреждения", null=True)
```

Рисунок 5 – Создание таблицы «tvsp»

Следующей была создана таблица «DiagnosticEqip» («Диагностическое оборудование»), где также есть поля типа Boolean.Внешним ключом для таблицы «DiagnosticEqip» является поле«tvsp», связывающее еес таблицей «tvsp». Создание таблицы представлено на рисунке 6.

```
class DiagnosticEqip(models.Model):
    diagnostic_id = models.IntegerField(
        u'ID диагностического оборудования',
        db_column='diagnostic_id',
        primary_key=True
    )

    name = models.CharField(u'Наименование оборудования',max_length=255,default = 'Заполните')
    type = models.CharField(u'Тип оборудования',max_length=255,default = 'Заполните')
    release_year = models.DateField(u'Год выпуска',null=True, blank=True)
    location = models.CharField(u'Место размещение оборудования',max_length=255,default = 'Заполните')
    lvs_connect_capasity = models.BooleanField(u'Наличие подключения к ЛВС?',default=False)
    research_year_number = models.CharField(u'Количество исследований в год',max_length=255,default = 'Заполните')
    average_size_research = models.CharField(u'Средний объем одного исследования (МБ)',max_length=255,default = 'Заполните')
    sign_activation_discom = models.BooleanField(u'Признак активации DICOM 3.0',default=False)
    existence_armd_discom = models.BooleanField(u'Наличие АРМ диагноста',default=False)

    tvsp = models.ForeignKey('tvsp', models.PROTECT,verbose_name=u"id ТВСП", null=True)
```

Рисунок 6 – Создание таблицы «DiagnosticEqip»

Затем была создана связующая таблица «Network»(«Сетевая инфраструктура»), содержащая два поля: внешний ключ «tvsp» и первичный ключ «network_id». На рисунке 7 показан фрагмент кода с созданием таблицы.

```
class Network(models.Model):
    network_id = models.IntegerField(
        u'ID сетевых параментров ТВСП',
        db_column='network_id',

        primary_key=True
    )

    tvsp = models.ForeignKey('tvsp', models.PROTECT,verbose_name=u"id ТВСП", null=True)
```

Рисунок 7 – Создание таблицы «Network»

На рисунке8 представлены программные кодысоздания таблиц, связанных с сетевыми настройками и сетевой безопасностью. Эти таблицы связаны с таблицей «Network» с помощью внешнего ключа «Network».

```python
class local_network(models.Model):
    local_network_id = models.IntegerField(
        u'ID локальной сети',
        unique=True,
        db_column='local_network_id',
        primary_key=True
        )

    number_port_lvs = models.CharField(u'Количество портов ЛВС', max_length=255,default = 'Заполните')
    network_equipment = models.CharField(u'Сетевое оборудование',max_length=255,default = 'Заполните')
    ip_connect = models.CharField(u'IP-адресация',max_length=255,default = 'Заполните')
    place = models.CharField(u'Площадь помещения', max_length=255, default='Заполните')
    server_room_compliance = models.CharField(u'Место размещения', max_length=255, default='Заполните')
    standart_rule = models.CharField(u' Соответствие требования к серверным помещениям', max_length=255,default='Заполните')
    LVS = models.BooleanField(u'Наличие схемы ЛВС', default=False)
    Network = models.ForeignKey('Network', models.PROTECT, verbose_name=u"Сетевая инфраструктура", null=True)

class security_network(models.Model):
    security_network_id = models.IntegerField(
        u'ID защищенной сети передачи данных',
        unique=True,
        db_column='security_network_id',
        primary_key=True
        )

    status_zcpd_connect = models.BooleanField(u'Статус подключения к ЗСПД',default = False)
    vipnet_coordinator = models.CharField(u'Технология подключения к ЗСПД',max_length = 255,default = 'Отсутствует')
    vipnet_hw1000 = models.BooleanField(u'ViPNet Coordinator',default = False)
    vipnet_hw100 = models.BooleanField(u'ViPNet HW1000',default = False)
    vipnet_client = models.BooleanField(u'iPNet Client',default = False)

    Network = models.ForeignKey('Network', models.PROTECT, verbose_name=u"Сетевая инфраструктура", null=True)

class network_connect(models.Model):
    network_connect_id = models.IntegerField(
        u'ID записи точки доступа в интернет',
        unique=True,
        db_column='network_connect_id',
        primary_key=True
        )

    status_interent_connect = models.CharField(u'Статус подключения к интернет',max_length=255,default = 'Заполните')
    tech_internet_connect = models.CharField(u'Технология подключения к интернет',max_length=255,default = 'Заполните')
    channel_capacity = models.CharField(u'Пропускная способность канала',max_length=255,default = 'Заполните')
    tariff_capacity = models.CharField(u'Пропускная способность по тарифу',max_length=255,default = 'Заполните')
    service_provider = models.CharField(u'Поставщик услуг',max_length=255,default = 'Заполните')
    cost_in_month = models.CharField(u'Стоимость услуг в месяц',max_length=255,default = 'Заполните')
    participation_in_fp = models.CharField(u'Участие в ФП',max_length=255,default = 'Заполните')

    Network = models.ForeignKey('Network', models.PROTECT, verbose_name=u"Сетевая инфраструктура", null=True)
```

Рисунок 8 – Программные коды создания таблиц «local_network», «security_network»и «network_connect»

Далее на рисунке 9 представлен программный код создания таблицы«Arm» («Оборудование АРМ»), которая аналогично предыдущим таблицам связана с таблицей «Network» с помощью внешнего ключа «Network». Помимо собственного ключа, в данную таблицу записывается номер кабинета, где расположено АРМ.

```
class Arm(models.Model):
    arm_id = models.IntegerField(
        u'Номер рабочего места',
        unique=True,
        db_column='arm_id',
        primary_key=True
        )
    number_room = models.CharField(u'Номер кабинета',max_length=255,default = 'Заполните')

    Network = models.ForeignKey('Network', models.PROTECT, verbose_name=u"Сетевая инфраструктура", null=True)
```

Рисунок 9 – Программный код создания таблицы «Arm»

Таблицы, создание которых представлено на рисунке10, содержат информацию об определенном рабочем месте: установленное программное обеспечение, характеристики системного блока, информацию о мониторе, компьютерной мыши и клавиатуре.

```
class Software(models.Model):
    software_id = models.AutoField(
        u'ID Программного обеспечения',
            unique=True,
            db_column='software_id',
            primary_key=True
        )
    name = models.CharField(u'Название',max_length=255,default = 'Заполните')
    developer = models.CharField(u'Разработчик',max_length=255,default = 'Заполните')
    home_soft = models.BooleanField(u'Наличие в реестре отечественного ПО')
    verison = models.CharField(u'Версия',max_length=255,default = 'Заполните')
    price = models.CharField(u'Цена',max_length=255,default = 'Заполните')
    purchase_year = models.DateField(u'Год покупки',null=True, blank=True)

    Arm = models.ForeignKey('Arm', models.PROTECT, verbose_name=u"АРМ", null=True)
```

Рисунок 10 – Программные коды создания таблиц «Software», «Mouse», «Keyboard», «Monitor»и «SystemBlock», лист 1

```
class Mouse(models.Model):
    mouse_id = models.AutoField(
        u'ID Компьютерной мышки',
            unique=True,
            db_column='mouse_id',
            primary_key=True
        )
    model = models.CharField(u'Модель', max_length=255,default = 'Заполните')
    inventory_number = models.CharField(u'Инвентарный номер',max_length=255,default = 'Заполните')
    purchase_year = models.DateField(u'Год покупки',null=True, blank=True)

    Arm = models.ForeignKey('Arm', models.PROTECT, verbose_name=u"АРМ", null=True)
class Keyboard(models.Model):
    keyboard_id = models.AutoField(
        u'ID Клавиатуры',
        unique=True,
        db_column='keyboard_id',
        primary_key=True
    )
    model = models.CharField(u'Модель', max_length=255,default = 'Заполните')
    inventory_number = models.CharField(u'Инвентарный номер',max_length=255,default = 'Заполните')
    purchase_year = models.DateField(u'Год покупки',null=True, blank=True)

    Arm = models.ForeignKey('Arm', models.PROTECT, verbose_name=u"АРМ", null=True)
class Monitor(models.Model):
    monitor_id = models.AutoField(
        u'ID Монитора',
        unique=True,
        db_column='monitor_id',
        primary_key=True
    )
    model = models.CharField(u'Модель', max_length=255,default = 'Заполните')
    diagonal = models.CharField(u'Диагональ',max_length=255,default = 'Заполните')
    inventory_number = models.CharField(u'Инвентарный номер',max_length=255,default = 'Заполните')
    purchase_year = models.DateField(u'Год покупки',null=True, blank=True)

    Arm = models.ForeignKey('Arm', models.PROTECT, verbose_name=u"АРМ", null=True)
```

Рисунок 10, лист 2

```
class SystemBlock(models.Model):
    sb_id = models.AutoField(
        u'ID системного блока',
        unique=True,
        db_column='sb_id',
        primary_key=True,
    )
    purchase_year = models.DateField(u'Дата покупки', null=True, blank=True)
    inventory_number = models.CharField(u'Инвентарный номер', max_length=255,default = 'Заполните')
    processor_model = models.CharField(u'Модель процессора',max_length=255,default = 'Заполните')
    model_ram = models.CharField(u'Модель ОЗУ', max_length=255,default = 'Заполните')
    ram_size = models.CharField(u'Объем ОЗУ', max_length=255,default = 'Заполните')
    storage_system_type = models.CharField(u'Тип системы хранения',max_length=255,default = 'Заполните')
    storage_system_size = models.CharField(u'Объем хранения',max_length=255,default = 'Заполните')
    operating_system = models.CharField(u'Операционная система',max_length=255,default = 'Заполните')

    Arm = models.ForeignKey('Arm', models.PROTECT, verbose_name=u"АРМ", null=True)
```

Рисунок 10, лист 3

На рисунках 11 и 12 показан интерфейс работы с базой данныхс помощью фреймворка Django.

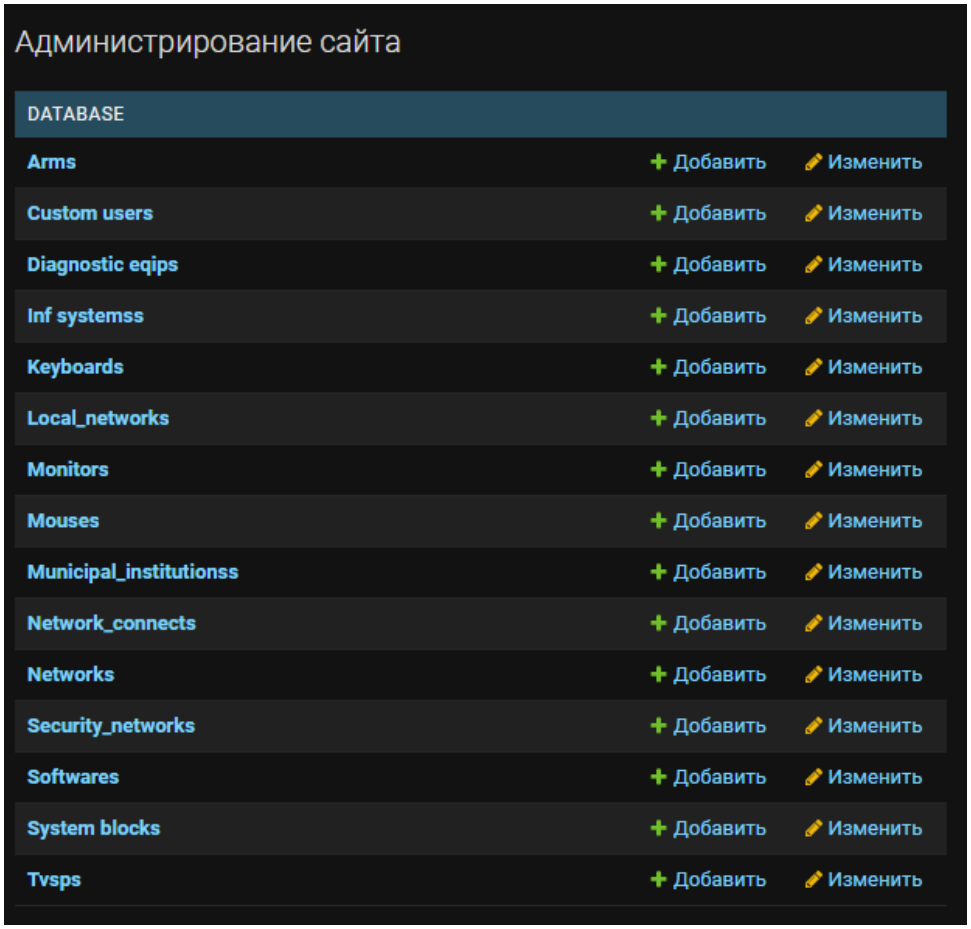

Рисунок 11 – Общий вид созданных таблиц

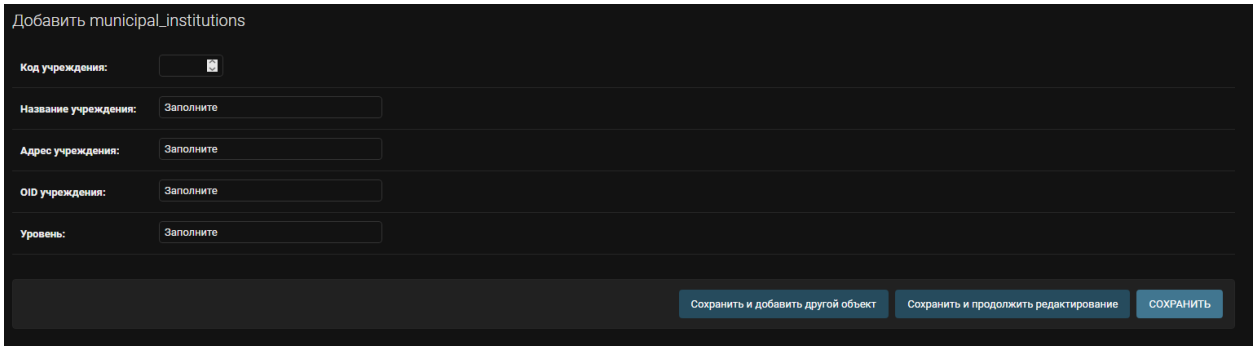

Рисунок 12 – Добавление данных в таблицу«municipal_institutions»

2.2 Создание html-страниц для вывода информации из базы данных на экран

Данные из базы данных выводятся на страницы веб-приложения. Разработка функционала веб приложения изложена в ВКР Озол Виктора Андреевича.

Вывод списка медицинских организаций был организован с помощью цикла «forelindata» (Рисунок 13).

```
<h3>Медицинская организация</h3>

    <p>Вам необходимо выбрать медицинскую организацию для получения информации о ней.</p>
    <ul class="list-group">
        {% for el in data  %}
        <li>
            <td>   <a href="{% url 'database' el.municipal_id %}"class="list-group-item list-group-item-action list-group-item-secondary">
                   №{{el.municipal_id}}: {{ el.name }}</a>
            </td>
        </li>
        {% endfor %}
    </ul>
```

Рисунок 13 – Фрагмент html-кода для страницы с выбором медицинской организации

Данные выводятся списком в виде ссылок, при нажатии на которые открывается дальнейшая информация. На рисунке 14 показан внешний вид созданной html-страницы.

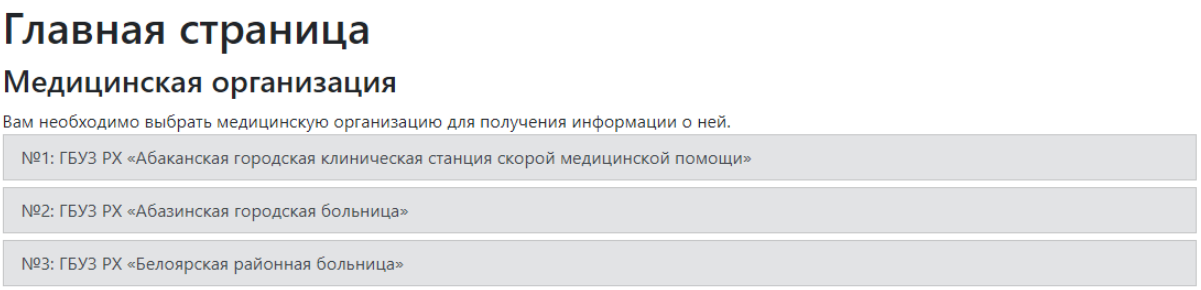

Рисунок 14 – Вид html-страницы со списком медицинских организаций

На главной странице пользователь выбирает нужную медицинскую организацию. При этом на следующие страницы передается соответствующий идентификатор.

Для вывода информации из выбранной таблицы по ключу для каждой таблицы написана подпрограмма-функция. Функции написаны для каждой отдельной страницы в файле views.py. Для корректной работы функции требовалась указать переменную, в которую запишется внешний ключ. Именно понему выводится нужная информация сначала о медицинской организации, потом о ТВСП этой организации, далее – информацию о сети внутри выбранного ТВСП и об АРМ. Далее определен путь,указывающий адрес html-страницы для соответствующей функции.

На главной странице информационной системы в кнопках с наименованиями медицинских организаций хранятся значения ключей municipal_id. Именно они и передаются в функции обработки следующих таблиц (переменная mo_id).

На рисунке 15приводятся программные коды для отбора информации из таблиц «municipal_institutions», «InfSystems» и «tvsp».

```
@login_required
def before_tvsp(request, mo_id):
    data = tvsp.objects.filter(municipal_institutions=mo_id).all()
    return render(request, 'database/before_tvsp.html', {'data':data, 'tvspid': mo_id})

@login_required
def infsystems(request, mo_id):
    data = InfSystems.objects.filter(municipal_institutions=mo_id).all()
    return render(request, 'database/infsystems.html', {'data':data})

@login_required
def maininfo(request, mo_id):
    data = municipal_institutions.objects.filter(pk=mo_id).all()
    return render(request, 'database/maininfo.html', {'data':data})

@login_required
def mo(request, mo_id):
    data = municipal_institutions.objects.all()
    return render(request, 'database/mo.html', {'data':data, 'mo_id':mo_id})

@login_required
def database(request, mo_id = 0):
    data = municipal_institutions.objects.filter(pk=mo_id).all()
    return render(request, 'database/mo.html', {'data':data, 'mo_id':mo_id})
```

Рисунок 15 –Объявление функций в файле views.рудля страниц с информацией о медицинских организациях, информационных системах и ТВСП

На рисунке 16 показан фрагмент html-кода для вывода общей информации о медицинской организации. В data записаны данные из таблицы «Основные сведения о медицинских организациях» с id медицинской организации, выбранным на предыдущей странице. Данные выводятся списком. Для создания списка использован тег «<ul>», а для создания элементов списка – тег «li». В двойных фигурных скобках записываются переменные, содержащие информацию из базы данных.

```
<h3>Медицинская организация</h3>

{% for el in data %}
<ul class = "list-group">
    <li class = "list-group-item active"> <p>№ {{ el.municipal_id }}: {{ el.name }} <a></a></p></li>
    <li class = "list-group-item ">

      <p class="text_card">  OID учреждения: </p> {{ el.oid_municipal }}
        <br> <p class="text_card">Адрес учреждения: </p> {{ el.address }}
        <br> <p class="text_card">Уровень учреждения: </p> {{ el.level_municipal }}
    </li>
</ul>
{% endfor %}
```

Рисунок 16 – Фрагмент html-кода для страницы с информацией о выбранной медицинской организации

На рисунке 17 представлена страница с информацией о выбранной медицинской организации.

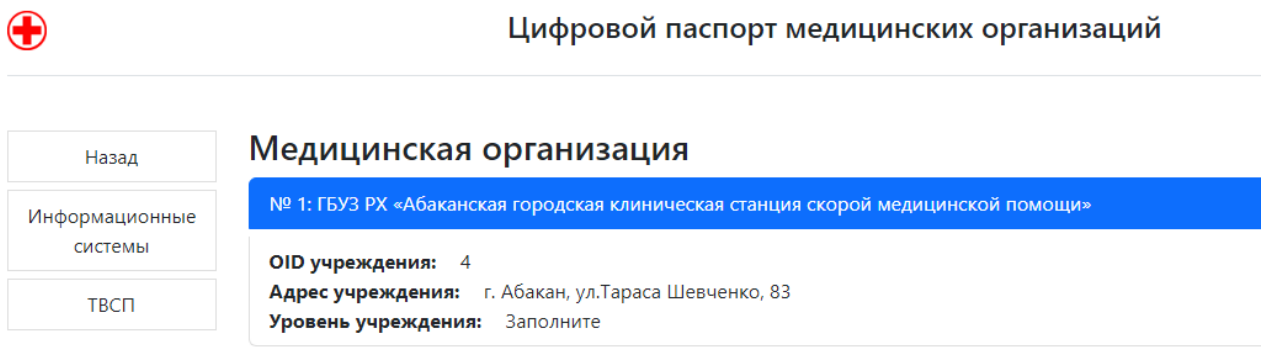

Рисунок 17 – Внешний вид страницы с информацией оГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

На рисунках 18 и 19 продемонстрированы html-страницы с информацией об информационных системах и ТВСП выбранной медицинской организации.

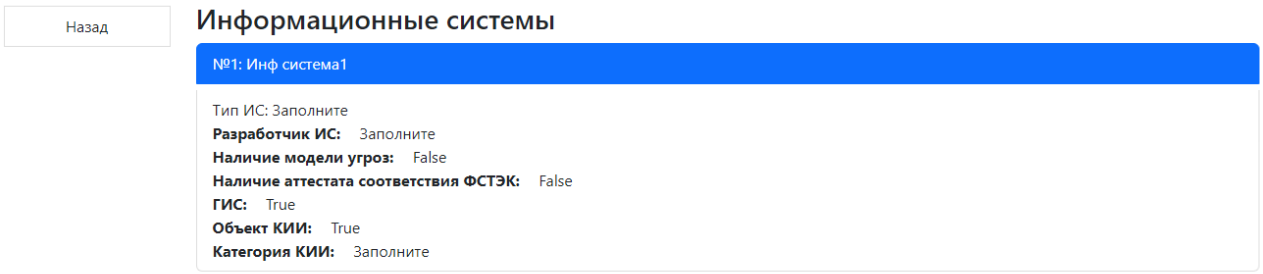

Рисунок 18 – Внешний вид страницы с информацией об информационных системахГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

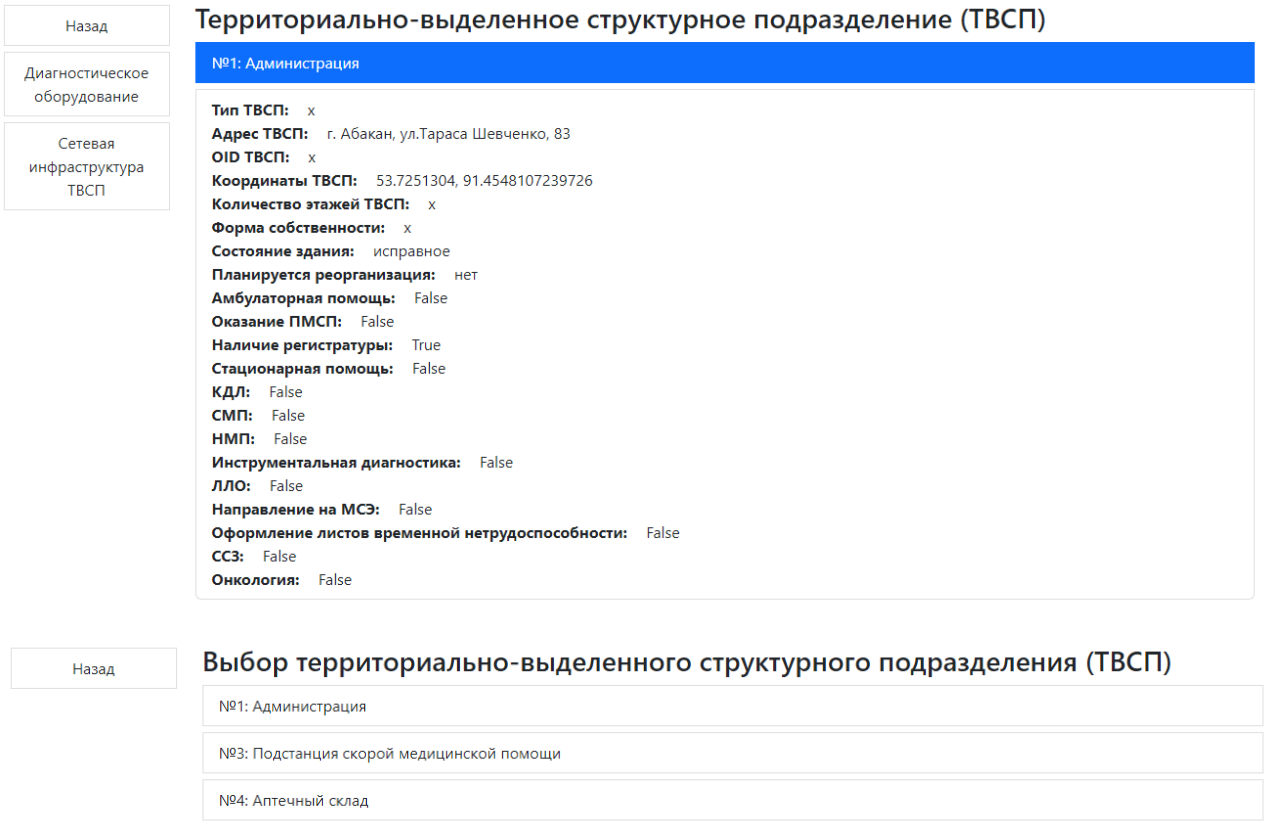

Рисунок 19 – Внешний вид страницы с информацией обадминистрации ГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

На следующем этапе по внешнему ключу из таблицы «tvsp» берется соответствующее значение переменнойtvspid, которое является внешним ключомпри отборе данных из таблиц «Network»и «DiagnosticEqip». На рисунке 20 показанысоответствующие функции.

```python
@login_required
def networks(request, tvspid):
    data = Network.objects.filter(network_id=tvspid).all()
    return render(request, 'database/network.html', {'data':data, 'networkid':tvspid})

@login_required
def before_networks(request, tvspid):
    data = Network.objects.filter(tvsp=tvspid).all()
    return render(request, 'database/before_network.html', {'data':data, 'networkid':tvspid})

@login_required
def diagnostic_eqips(request, tvspid):
    data = DiagnosticEqip.objects.filter(tvsp=tvspid).all()
    return render(request, 'database/diagnostic_eqip.html', {'data':data})

@login_required
def tvsps(request, mo_id):
    data = tvsp.objects.filter(tvsp_id=mo_id).all()
    return render(request, 'database/tvsp.html', {'data':data, 'tvspid': mo_id})
```

Рисунок 20 – Объявление функций в файле views.рудля страниц с информацией о диагностическом оборудовании и сетевой инфраструктуре

Аналогично предыдущим страницам был организован вывод соответствующей информации (Рисунки 21, 22).

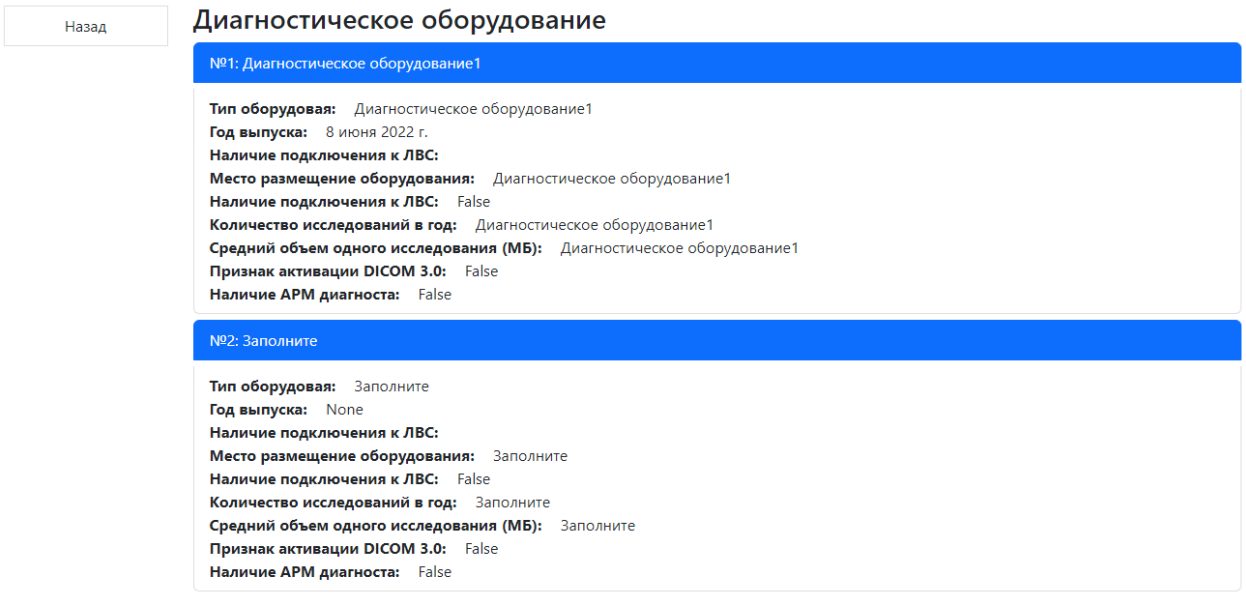

Рисунок 21 – Внешний вид страницы с информацией о диагностическом оборудовании администрацииГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

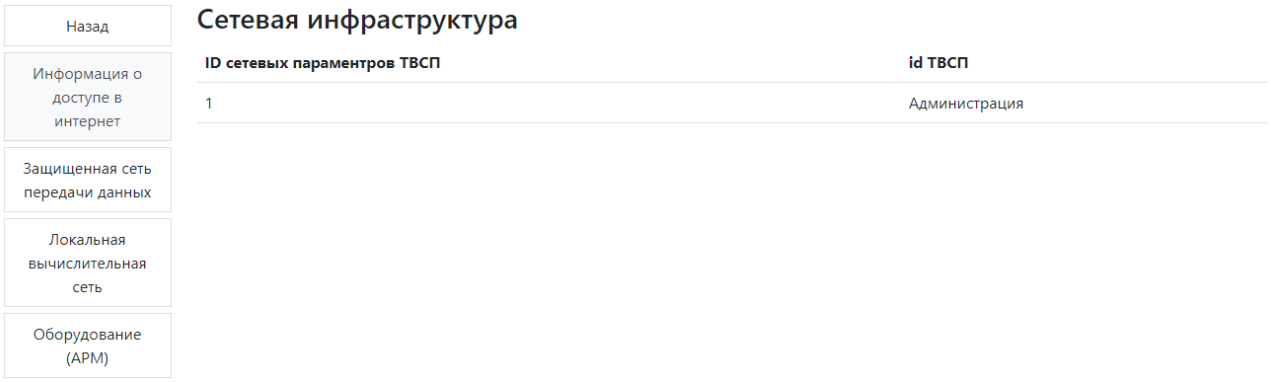

Рисунок 22 – Внешний вид страницы с информацией о сетевой инфраструктуре администрацииГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

На следующем этапе по внешнему ключу из таблицы «Network» берется соответствующее значение переменнойnetworkid, которое является внешним ключомпри отборе данных из таблиц «Security_network», «local _network»и «network_connect» (Рисунок 23).

```
@login_required
def security_networks(request, networkid):
    data = security_network.objects.filter(Network=networkid).all()
    return render(request, 'database/security_network.html', {'data':data})

@login_required
def network_connects(request, networkid):
    data = network_connect.objects.filter(Network=networkid).all()
    return render(request, 'database/network_connect.html', {'data':data})

@login_required
def arms(request, networkid):
    data = Arm.objects.filter(arm_id=networkid).all()
    return render(request, 'database/arm.html', {'data':data, 'armid':networkid})

@login_required
def before_arms(request, networkid):
    data = Arm.objects.filter(Network=networkid).all()
    return render(request, 'database/before_arm.html', {'data':data, 'armid':networkid})

@login_required
def local_networks(request, networkid):
    data = local_network.objects.filter(Network=networkid).all()
    return render(request, 'database/local_network.html', {'data':data})
```

Рисунок 23 – Объявление функций в файле views.рудля страниц с информацией о защищенности сети, о локальной вычислительной сети, АРМ и подключении к сетиИнтернет

Страницы с указанной информацией представлены на рисунках 24–27.

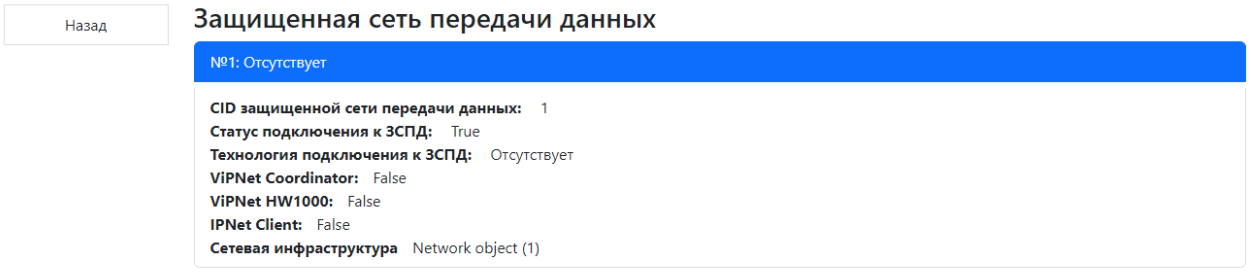

Рисунок 24 – Внешний вид страницы с информацией о защищенности сети1 администрации ГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

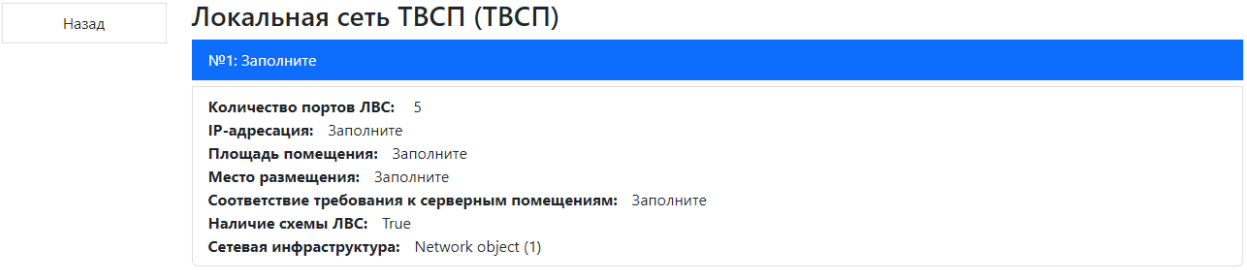

Рисунок 25 – Внешний вид страницы с информацией о локальной вычислительной сети1 администрации ГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

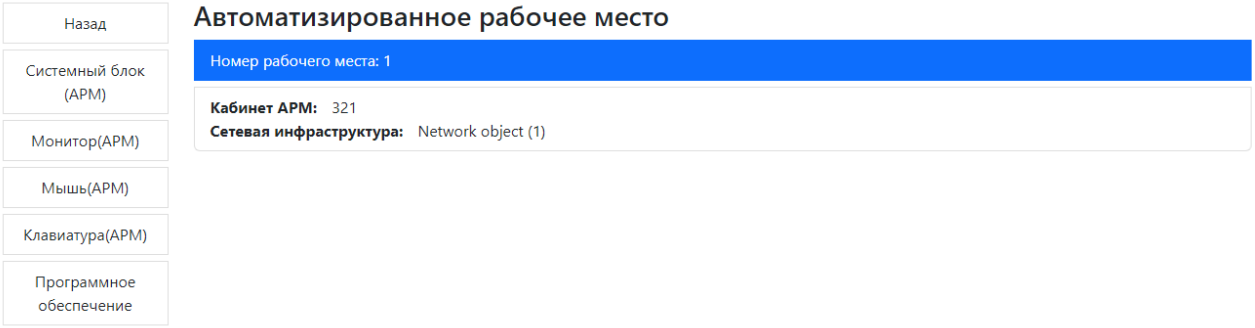

Рисунок 26 – Внешний вид страницы с информацией обАРМ кабинета 321 администрации ГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

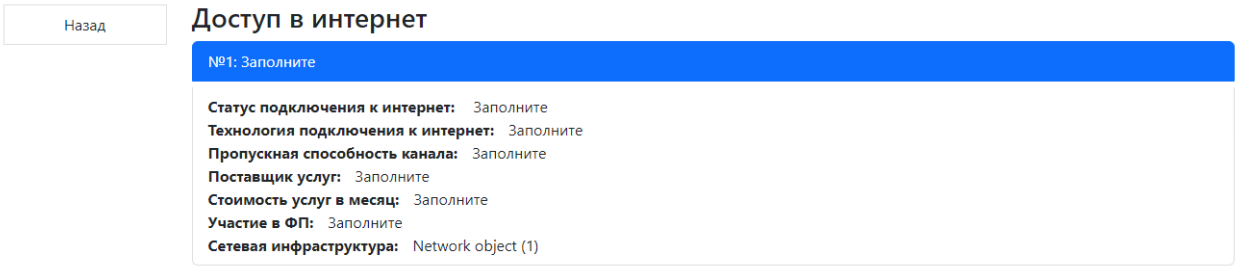

Рисунок 27 – Внешний вид страницы с информацией оподключении к сети Интернетв кабинете 321 администрации ГБУЗ РХ «Абаканская городская клиническая станция скорой медицинской помощи»

Для остальных таблиц внешним ключом служит переменная armidиз поля arm_idтаблицы «Arm» (Рисунок 28).

```
@login_required
def mouses(request, armid):
    data = Mouse.objects.filter(Arm=armid).all()
    return render(request, 'database/mouse.html', {'data':data})

@login_required
def monitors(request, armid):
    data = Monitor.objects.filter(Arm=armid).all()
    return render(request, 'database/monitor.html', {'data':data})

@login_required
def system_block(request, armid):
    data = SystemBlock.objects.filter(Arm=armid).all()
    return render(request, 'database/system_block.html', {'data':data})
@login_required
def softwares(request, armid):
    data = Software.objects.filter(Arm=armid).all()
    return render(request, 'database/software.html', {'data':data})

@login_required
def keyboards(request, armid):
    data = Keyboard.objects.filter(Arm=armid).all()
    return render(request, 'database/keyboard.html', {'data':data})
```

Рисунок 28 – Объявление функций в файле views.рудля страниц с информацией о компьютерной мыши, мониторе, системном блоке, клавиатуре и программном обеспечении

Страницы с указанной информацией представлены на рисунках 29–33.

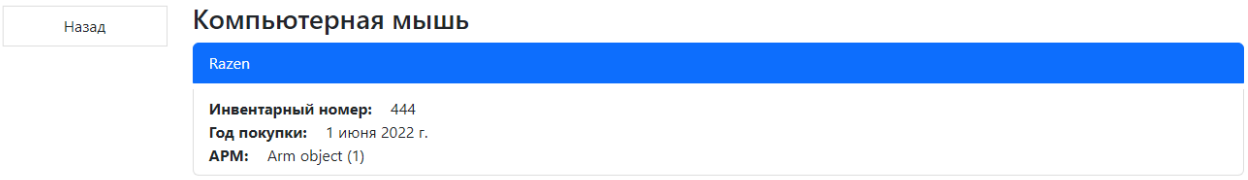

Рисунок 29 – Внешний вид страницы с информацией о компьютерной мышив кабинете 321

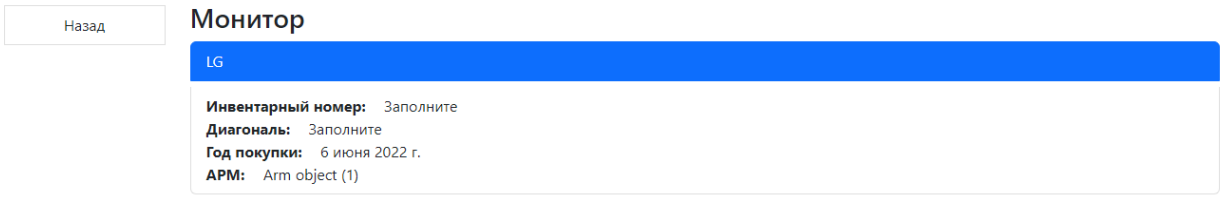

Рисунок 30 – Внешний вид страницы с информацией о мониторев кабинете 321

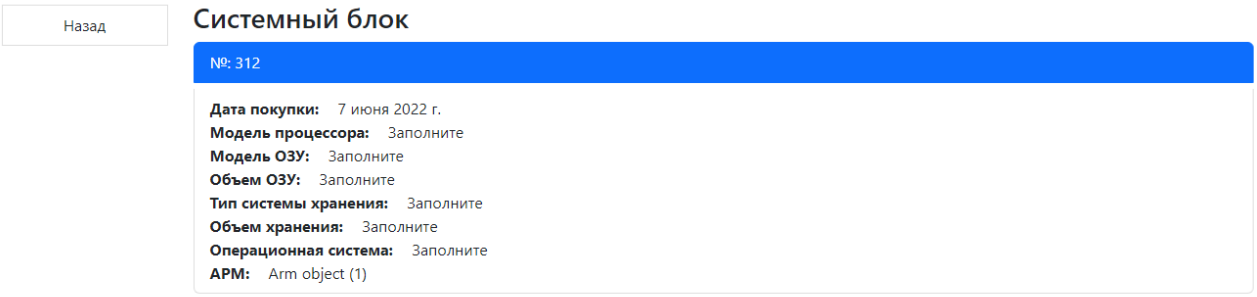

Рисунок 31 – Внешний вид страницы с информацией о системном блокев кабинете 321

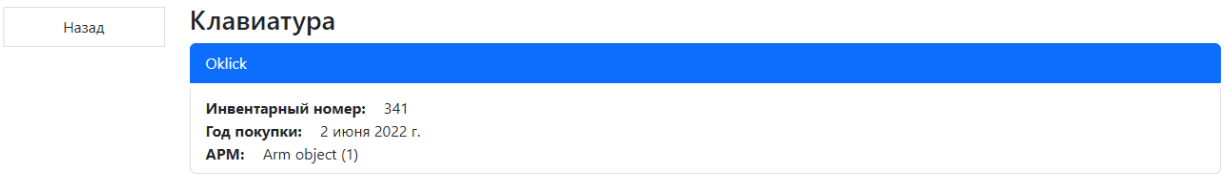

Рисунок 32 – Внешний вид страницы с информацией о клавиатуре в кабинете 321

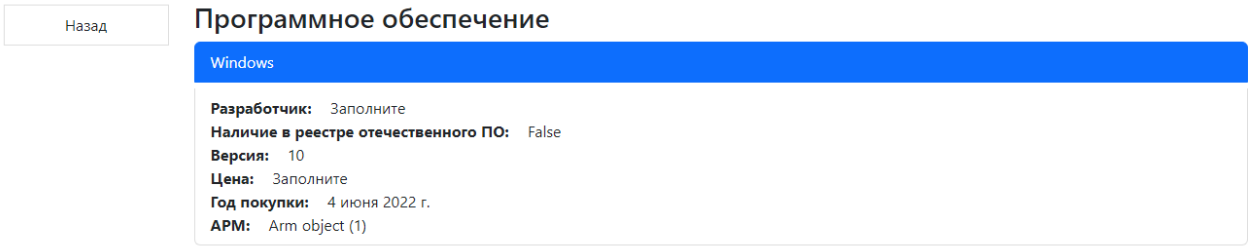

Рисунок 33 – Внешний вид страницы с информацией о программном обеспечениикомпьютера в кабинете 321

После создания функций в файле url.рунужно прописать пути перехода по определенным ссылкам.Также в пути объявляются переменные, используемые при создании функций.Это нужно для того, чтобы при переходе на новую страницу в адресеучитывался и выбранный ключ, как показано на рисунке 34.

```
urlpatterns = [
    path('software/<int:armid>/', views.softwares, name = 'software'),
    path('keyboard/<int:armid>/', views.keyboards, name = 'keyboard'),
    path('mouse/<int:armid>/', views.mouses, name = 'mouse'),
    path('monitor/<int:armid>/', views.monitors, name = 'monitor'),
    path('system_block/<int:armid>/', views.system_block, name = 'system_block'),
    path('arm/<int:networkid>/', views.arms, name = 'arm'),
    path('before_arm/<int:networkid>/', views.before_arms, name = 'before_arm'),
    path('local_network/<int:networkid>/', views.local_networks, name = 'local_network'),
    path('security_network/<int:networkid>/', views.security_networks, name = 'security_network'),
    path('network_connect/<int:networkid>/', views.network_connects, name = 'network_connect'),
    path('network/<int:tvspid>/', views.networks, name = 'network'),
    path('before_network/<int:tvspid>/', views.before_networks, name = 'before_network'),
    path('diagnostic_eqip/<int:tvspid>/', views.diagnostic_eqips, name = 'diagnostic_eqip'),
    path('tvsp/<int:mo_id>/', views.tvsps, name = 'tvsp'),
    path('infsystems/<int:mo_id>/', views.infsystems, name = 'infsystems'),
    path('maininfo/<int:mo_id>/', views.maininfo, name = 'maininfo'),
    path('before_tvsp/<int:mo_id>/', views.before_tvsp, name = 'before_tvsp'),
    path('<int:mo_id>/', views.database, name = 'database'),
    path('', views.database, name = 'database'),
]
```

Рисунок 34 – Добавление пути для перехода по ссылкам

2.3 Создание html-страниц для вывода отчета о заполненности сведений в таблицах

Для создания отчета о незаполненных строках в таблицах в отдельной папке был создан проект report_info. Внутри папки в файле views.рудля каждой таблицы были описаны функции,в которых находится количествонезаполненных строк. Так как значением по умолчанию в каждом поле является слово «Заполните», тоналичие именно этого значенияявляется условием отбора незаполненной ячейки. На рисунке 35 показана описываемая функция,в которойнаходится количество незаполненных строк в таблице «municipal_institutions».

```
def municipal(mo_id):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = " SELECT * from database_municipal_institutions "
    Municipal.execute (sqlite_select_query)
    record = Municipal.fetchall()
    row = record[mo_id-1]
    i = 0
    if 'Заполните' in set(row) and row[0]==mo_id:
        i=i+1

    return  i
```

Рисунок 35 – Создание функции для вывода количества незаполненных строк в таблице «municipal_institutions»

В первую очередь с помощью sql-запросов идет подключение к базе данных.Затем, используя команду **Select * fromdatabase_municipal_institutions**производится выбор всех данных из таблицы «municipal_institutions» в переменную «record».С помощью условия **‘Заполните’instr(row)** выбранная строка проверяется на наличие в ней

слова«Заполните», а условие**row[0]==mo_id**, нужно для проверки idмедицинской организации, чтобы данные считались только для нее.

Для следующей функции, подсчитывающей количество незаполненных строк в информационной системе, дописали еще одно условие, т.к. проверяются информационные системы, входящие в определённую медицинскую организацию. На рисунке 36 показано создание функции для вывода информации о количестве незаполненных строк в таблице «InfSystems» системы».

```
def InfSystems_info(mo1):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_infsystems """
    Municipal.execute(sqlite_select_query)
    record1 = Municipal.fetchall()
    g, g1 = 0, 0
    for row1 in record1:
        if row1[9]==mo1:
            g1 += 1
            if 'Заполните' in set(row1):
                g = g + 1

    data_none = str(g)
    data_none1 = str(g1)
    return data_none, data_none1
```

Рисунок 36 – Функция для вывода количества незаполненных строк в таблице «InfSystems»

Программные коды функцийдля оставшихся таблиц были написаны аналогичным образом. Фрагменты кодов для остальных страниц представленыв приложении А.

Функция для сбора всей найденной информации представлена на рисунке 37.Затем вышеописанные функции были объявлены в новой функции «report» для подсчета общего количества строк и количества незаполненных строк.

```
def report(request, mo_id):
    if request.user.is_authenticated == True:
        if request.user.is_superuser == True:
            data = municipal_institutions.objects.all()
        else:
            mo_filter = CustomUser.objects.filter(mun_admin_id=request.user.id).first()
            if mo_filter:
                data = municipal_institutions.objects.filter(municipal_id=mo_filter.municipal_institutions_id).all()

        data_municipal = municipal(mo_id)
        data_municipal1 = InfSystems_info(mo_id)
        data_municipal2 = tsvp_info(mo_id)
        date_de = diagnostic_eqip(tvspid)
        data_network = network_tvsp(tvspid)
        data_security_network = security_network(networkid)
        data_network_connect = network_connect(networkid)
        data_local_network = local_network(networkid)
        data_arm = arm(networkid)
        data_keyboard = keyboard(armid)
        data_mouse = mouse(armid)
        date_monitor = monitor(armid)
        date_systemblock = systemblock(armid)
        date_software = software(armid)
        data_ob = int(data_municipal) + \
                  int(data_municipal1[0]) + \
                  int(data_municipal2[0]) + \
                  int(date_de[0]) + \
                  int(data_network) +\
                  int(data_security_network[0]) + \
                  int(data_network_connect[0]) + int(data_local_network[0]) + \
                  int(data_arm[0]) + int(data_keyboard[0]) + int(data_mouse[0]) + \
                  int(date_monitor[0]) + int(date_systemblock[0]) + int(date_software[0])
        data_vsego = 1 + int(data_municipal1[1]) + int(data_municipal2[1]) + \
                     int(date_de[1]) + int(data_security_network[1]) + int(data_network_connect[1]) + \
                     int(data_local_network[1]) + int(data_arm[1]) + int(data_keyboard[1]) + int(data_mouse[1]) + \
                     int(date_monitor[1]) + int(date_systemblock[1]) + int(date_software[1])
```

Рисунок 37 – Программный код функции для подсчета общего количества строк и количества незаполненных строк

На рисунке 38 показан вид html-страницы с отчетом о заполненности информации в таблицах по выбранной медицинской организации.

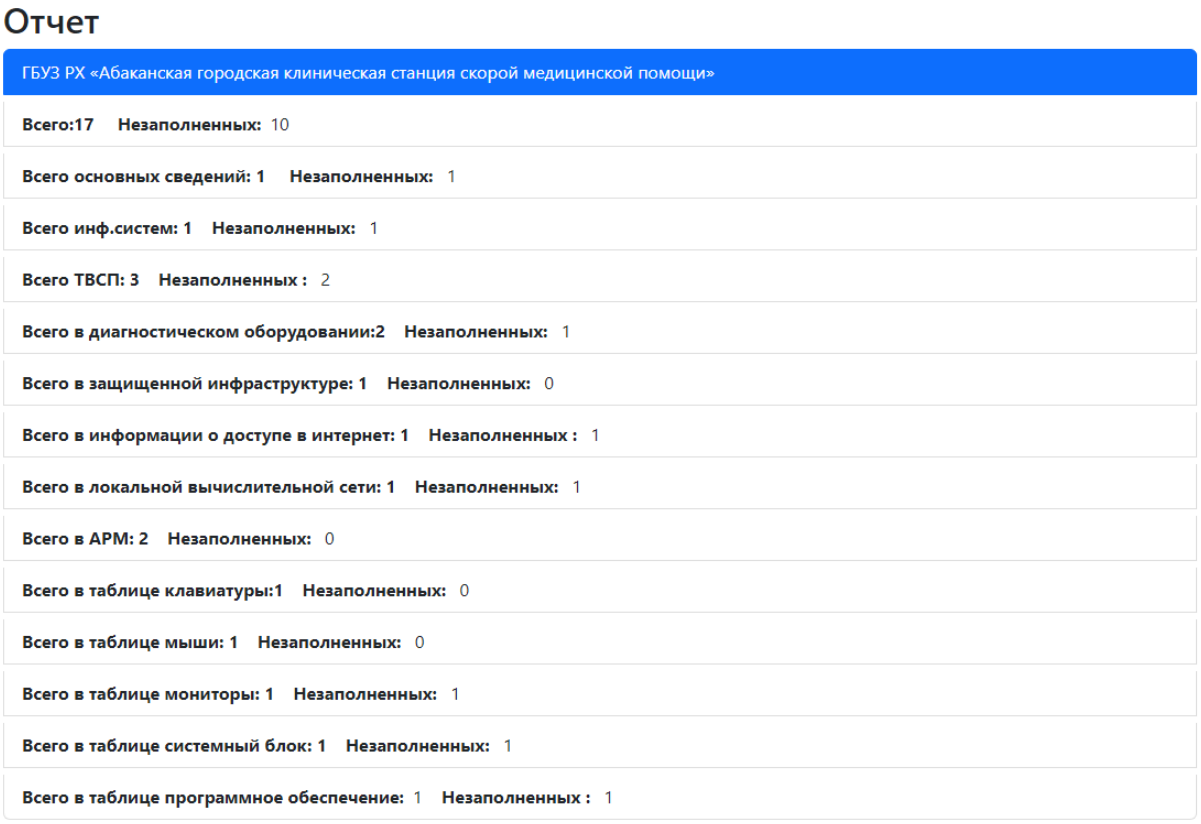

Рисунок 38 – Вид html-страницы с информацией о заполненности данных дляГБУЗ РЗ «Абаканская городская клиническая станция скорой медицинской помощи»

2.4Реализация выбора информации из базы данных по условиям пользователя

База данных содержит большой объем информации, поэтому при просмотре пользователю удобно самому определять условия отбора. При разработке веб-приложений эта задача решается при помощи фильтров.

Для любой таблицы в первую очередь организован фильтр по наименованию медицинской организации. Так как список медицинских организаций является строго определенным, то для этого условия на html-странице было создано поле с выпадающим списком (Рисунок 39).

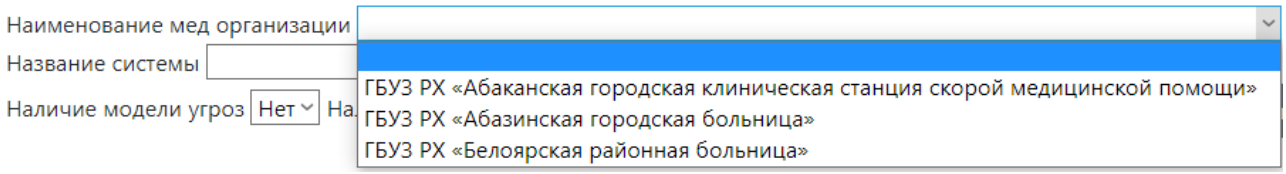

Рисунок 39 – Поле с выпадающим списком для выбора наименования медицинской организации

При создании фильтров необходимо учитывать тип полей в соответствующей таблице базы данных. Рассмотрим организацию фильтра для таблицы «InfSystems». Таккакнаименования информационных систем определены заранее, то поле «name» является полем с выпадающим списком;поля «type», «developer», «cii_category» являются текстовыми полями; поля «treat_model_presence», «conformity_certificate», «object_cii»и «gis» имеют тип Boolean (Рисунок 40).

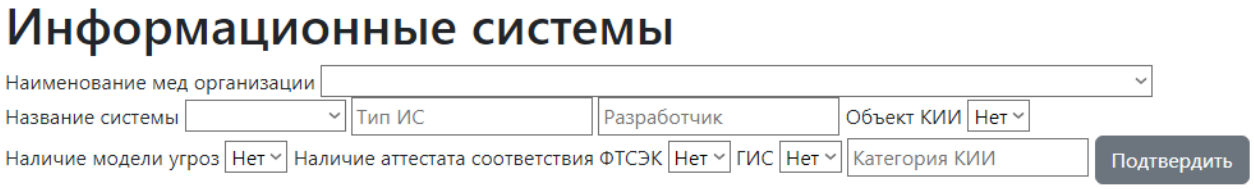

Рисунок 40 – Поля для записи условий отбора

Информация из указанных полей записывается в переменные, значения которых передаются в функцию.

До передачи данных из форму в функцию необходимо инициализировать переменные для корректной работы функции и отображения информации на странице, особенно это важно для значений типа Boolean, иначе при переходе на страницу возникнет ошибка о несоответствии типов, т.к передастся пустая строка.

Программный код функции представлен на рисунке 41.

```python
def filter_infsystems(request):

    nameinf = ''
    typeinf = ''
    developerinf = ''
    cii_categoryinf =''
    moinf =''
    treat_model_presenceinf = False
    conformity_certificateinf = False
    gisinf = False

    data1=InfSystems.objects.all()
    data2=list(municipal_institutions.objects.all())
    row = ''
    object_ciiinf = False
    if request.POST:
        moinf = request.POST['moinf']
        nameinf = request.POST['nameinf']
        typeinf = request.POST['typeinf']
        developerinf = request.POST['developerinf']
        object_ciiinf = request.POST['object_ciiinf']
        treat_model_presenceinf = request.POST['treat_model_presenceinf']
        conformity_certificateinf =request.POST['conformity_certificateinf']
        gisinf = request.POST['gisinf']
        cii_categoryinf = request.POST['cii_categoryinf']

    if moinf !='':
        for row in data2:
            if row.name==moinf:
                data = InfSystems.objects.filter(municipal_institutions=row.municipal_id)
    else:
        data = InfSystems.objects.filter(treat_model_presence=treat_model_presenceinf,
                                         conformity_certificate=conformity_certificateinf, gis=gisinf,
                                         object_cii=object_ciiinf)
    if nameinf != '':
        data = data.filter(name=nameinf,
                           treat_model_presence = treat_model_presenceinf,
                           conformity_certificate = conformity_certificateinf,
                           gis = gisinf, object_cii = object_ciiinf)


    if typeinf != '':
        data = data.filter(type=typeinf)

    if developerinf != '':
        data = data.filter(developer=developerinf)

    if cii_categoryinf != '':
        data = data.filter(cii_category=cii_categoryinf)

    if nameinf == '' and typeinf == '' and developerinf == '' \
            and object_ciiinf == False and \
            treat_model_presenceinf == False and \
            conformity_certificateinf == False and gisinf == False:
        data = InfSystems.objects.all()

    return render(request, 'filter/filter_infsystem.html', {'data':data,'data1':data1, 'data2':data2})
```

Рисунок 41 – Программный код функции для фильтрации данных в таблице «InfSystems»

Объект data1 в представленном коде является объектом типа QuerySetи отвечает за вывод наименований информационных систем в поле отбора.

Объект data2 – список; необходим для определения идентификатора медицинской организации. Элементами списка являются объекты QuerySet (в программном коде им соответствует переменная row). Тогда наименование медицинской организации можно считать с помощью точечной нотации: row.name.Объект data – QuerySet; необходим для вывода информации из базы данных на html-страницу.Так как введенные пользователем условия отбора позволяют выполнить фильтр данных, а затем эти данные должны быть переданы обратно на сайт, то для этого понадобился Post-запрос. Если пользователь оставит поля отбора незаполненными, то на экране увидит всю информацию из таблицы информационные системы. Примеры реализации пользовательских фильтров представлены на рисунках 42, 43

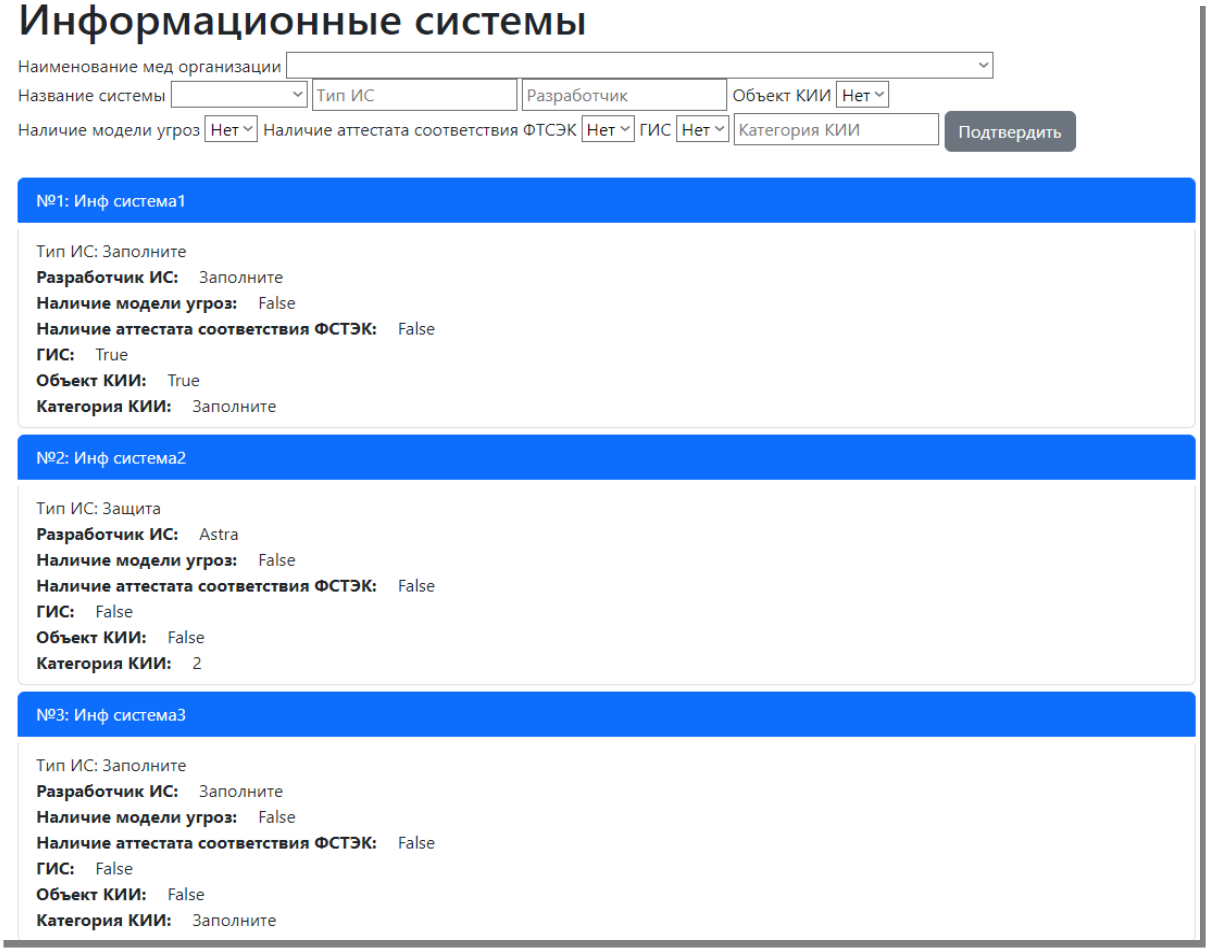

Рисунок 42 – Пример вывода информации при незаполненных фильтрах отбора

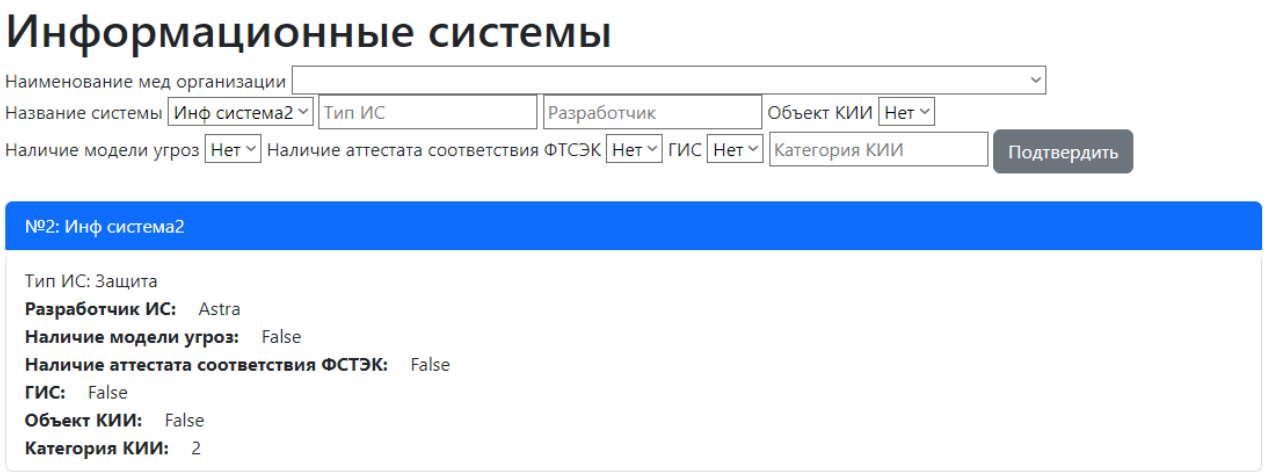

Рисунок 43 – Пример вывода информации при заполненных фильтрах отбора

2.5 Выводы по разделу Разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

В результате была создана информационная система,позволяющая хранить и обрабатывать информацию о медицинских организациях. Также эта система позволяет отображать актуальную информацию о заполненности базы данных. Пользователь может просмотреть информацию о выбранной медицинской организации, а также просмотреть информацию из определенной таблицы при необходимых условиях отбора.

3 Расчёт затрат реализации проекта «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

3.1 Расчет проектных затрат

Первоначально необходимо рассчитать проектные затраты. Проектные затраты рассчитываются по формуле

$$К_{пр} = К_{зп} + К_{ипс} + К_{свт} + К_{проч}, \quad (1)$$

где $К_{зп}$ – затраты за заработную плату проектировщика ИС;

$К_{ипс}$ – затраты на инструментальные ПО для проектирования ИС;

$К_{свт}$ – затраты на средства вычислительной техники для проектирования ИС;

$К_{проч}$ – прочие затраты.

Минимальный размер оплаты труда в Абакане равен 13890 руб., также необходимо учесть северный и районный коэффициенты. Расчет заработной платы за месяц представлен в таблице 13.

Таблица 13 – Заработная плата разработчика ИС

Вид дохода	**Сумма, руб.**
Оклад	13890
Северный коэффициент	4167
Районный коэффициент	4167
Итог	22 224

Для оценки затрат на заработную плату нужно учитывать отчисления во внебюджетные фонды, которые составляют 30,2% от начисленной зарплаты:

$К_{зп} = 22224 \cdot 1{,}302 = 28935$ рублей за месяц, для одного разработчика.

Для двух разработчиков:

$К_{зп} = 28935 \cdot 2 = 57870$ рублей.

В таком случае затраты на оплату труда программиста за 1 день составят:

$К_{зп/1} = \frac{28935}{24} = 1206$ рублей за день, для одного разработчика.

Так как разработчика два, то

$К_{зп/2} = 1206 \cdot 2 = 2412$ рублей за день, для двух разработчиков.

Прочие затраты – затраты на непредвиденные расходы.

Для создания системы потребуются программные средства, представленные в таблице 14.

Таблица 14 – Необходимые программные средства

Название ПО	**Цена, руб.**
SQLite	Бесплатно
Pycharm	Бесплатно
Операционная система Windows 10	6970
Django	Бесплатно

Программное обеспечение Windows 10 покупается разово и не ограничено временем доступного пользования.

Используя перечень ресурсов из таблицы 14, можно вычислить месячную амортизацию по формуле

$$А_{мес} = С_т \cdot Н_а, \quad (2)$$

где $С_т$ – стоимость программных средств;

$Н_а$ – норма амортизации.

$$Н_А = \frac{1}{нсл} \cdot 100\% \quad (3)$$

где нсл – нормативный срок службы.

Амортизация вычисляется только для Windows 10, т.к. остальные программные продукты бесплатные.

Срок службы: 4 года (48 месяцев). Тогда получим:

$\mathrm{H_A} = \frac{1}{48} \cdot 100\% = 2{,}08$ за месяц.

Ежемесячная амортизация: $6970 \cdot 2{,}08\% = 145$ рублей.

Амортизация за 4 месяца: $145 \cdot 4 = 580$ рублей.

Затраты на инструментальные ПО для проектирования ИС: $\mathrm{К_{ипс}} = 580$ рублей.

Далее идет расчет амортизации для технического оборудования, используя формулы (3) и (4), где вместо стоимости программных средств используется общая стоимость оборудования. Список и стоимость оборудования представлены в таблице 15.

Таблица 15 – Перечень необходимого технического оборудования для разработки

Комплектующие	**Название**	**Количество**	**Цена, руб.**	**Срок службы, лет**
Процессор	AMD Ryzen 5 3400G, OEM	1	11590	4
Материнская плата	ASUS PRIME B450M-A	1	5780	4
Видеокарта	GTX 1660 super	1	19000	5
Кулер	DEEPCOOL GAMMAXX 300 FURY	1	1370	4
Оперативная память	KINGSTON HyperX FURY HX430C15FB3K2/16 DDR4—8ГБ	2	7490	6
SSD накопитель	KINGSTON A2000 SA2000M8/500G 500ГБ	1	5890	4
Блок питания	QDION Q-DION QD550, 550Вт	1	2800	5
Корпус	ATX FORMULA CL-3301B TG, черный	1	2640	4
Монитор	18.5" Монитор HP 19ka [T3U81AA]		5490	4
Мышь	Defender Patch MS-759	1	199	4
Клавиатура	RITMIX RKB-100	1	299	4
Итог			58358	

Срок службы компьютерных комплектующих составляет около 4 лет.

Срок эсплуатации:4 года (48 месяцев).

Норма амортизации: $\text{Н}_\text{А} = \frac{1}{48} \cdot 100\% = 2{,}08$ за месяц.

Ежемесячная амортизация: $58358 \cdot 2{,}08\% = 1213$ рублей.

Амортизация за период работы (4 месяца): $1213 \cdot 4 = 4852$ рубля, для одного разработчика.

Амортизация за период работы (4 месяца) для двоих: $4852 \cdot 2 = 9704$ рубля.

Прочие затраты на проектирование составляют 3% от суммы расходов на заработную плату, ПО и вычислительную технику.

Расчет проектных затрат представлен в таблице 16.

Таблица 16 – Расчет проектных затрат

Наименование затрат	**Затраты, руб**
К зп	57870
К ипс	580
К свт	9704
К проч	2044
Итог	70198

$\text{К}_{\text{пр}} = 70198$ рублей.

3.2 Расчет капитальных затрат

Капитальные затраты рассчитываются по формуле

$$\text{К} = \text{К}_{\text{пр}} + \text{К}_{\text{тс}} + \text{К}_{\text{лс}} + \text{К}_{\text{по}} + \text{К}_{\text{ио}} + \text{К}_{\text{об}} + \text{К}_{\text{оэ}} + \text{К}_{\text{проч}}, \quad (4)$$

где $\text{К}_{\text{пр}}$–затраты на проектирование ИС;

$К_{тс}$–затраты на технические средства управления;

$К_{лс}$–затраты на создание локальных связей;

$К_{пс}$–затраты на программные средства;

$К_{ио}$–затраты на формирования информационной базы;

$К_{об}$–затраты на обучение персонала;

$К_{оэ}$–затраты на опытную эксплуатацию;

$К_{проч}$ – прочие расходы.

$К_{тс}$=1206 руб., потому что ИС взаимодействует с другими системами, которые используются на предприятии и на настройку их работы уйдет 1 рабочий день расчёт затрат приведен ниже;

$К_{лс}$= 0, так как локальная сеть на предприятии работает и применяется для системы редко;

$К_{пс}$=0, т.к. программные средства для работы с системой предустановлены на компьютере администратора системы;

Для формирования базы данных потребуется не менее 7 рабочих дней:

$$К_{ио} = 7 \cdot 1206 = 8442 \text{ рубля},$$

$К_{об}$= 0, проведение обучения персонала для работы с системой не потребует много времени, поэтому не учитывается;

Заработная плата разработчика за 3 день внесения правок в проект в период эксплуатации:

$$К_{оэ} = 1206 \cdot 3 = 3618 \text{ рублей}.$$

$К_{проч}$ составляет 3% от суммы капитальных затрат.

По формуле 4 вычислены капитальные затраты:

$$К = 70198 + 1206 + 8442 + 1460 = 84924 \text{ рубля}.$$

Список капитальных затрат представлен в таблице 17. Структура капитальных затрат представлена на рисунке 44.

Таблица 17 – Капитальные затраты

Наименование затрат	**Затраты, руб.**
$К_{пр}$	70198
$К_{тс}$	1206
$К_{лс}$	0
$К_{пс}$	0
$К_{ио}$	8442
$К_{об}$	0
$К_{оэ}$	3618
$К_{проч}$	1460
Итог	84924

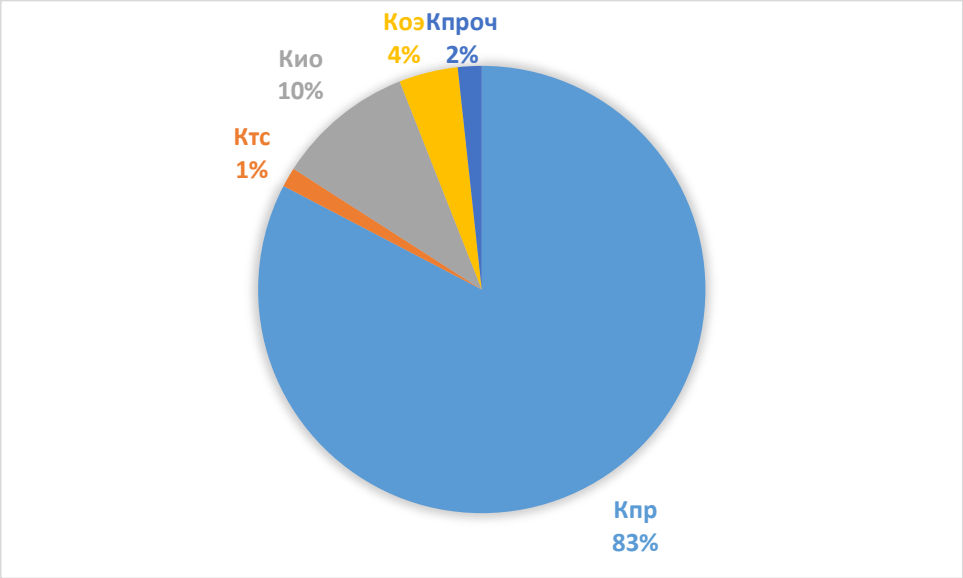

Рисунок 44 – Структура капитальных затрат

Из диаграммы видно, что большую часть составляют затраты на проектирование ИС.

3.3 Расчет эксплуатационных затрат

Расчет эксплуатационных затрат производится по формуле

$$C = C_{зп} + C_{ао} + C_{то} + C_{лс} + C_{ни} + C_{проч}, \quad (5)$$

где $C_{зп}$ – зарплата персонала, работающего с ИС;

$C_{ао}$ – амортизационные отчисления;

$C_{то}$ – затраты на техническое обслуживание

$C_{лс}$ – затраты на использования глобальных сетей

$C_{ни}$ – затраты на носители информации;

$C_{проч}$ – прочие затраты.

Предстоит работа по обновлению информации в базе данных и формированию отчетов составит не более 2 часов в месяц, что соответствует зарплате:

$C_{зп} = \frac{3412}{8} \cdot 2 \cdot 12 = 7248$ рублей; для двух разработчиков.

$C_{ао}$= 0, для работы базы данных необходим сервер. Ресурсы имеющегося сервера позволяют разместить на нем базу данных, без значительного увеличения нагрузку на сервер, т.е нет нужды покупать новый.

$C_{то} = 7248 \cdot 4$=28992 рубля, расходы для возможной доработки системы, 4 дня в год.

$C_{лс}$ = 0, для работоспособности системы потребуется интернет, но т.к. он оплачен в ГБУЗ РХ «РМИАЦ» для поддержания других систем, то учитывать в расходах на поддержку разрабатываемой системы не требуется.

$C_{ни}$ = 0, т.к. информация передается через интернет и хранится на сервере.

$C_{проч}$ составляет 3% от суммы эксплуатационных затрат;

Используя формулу (5) для расчета эксплуатационных затрат было получено:

$$C = (7248 + 28992) \cdot 1{,}03 = 37327 \text{ рублей.}$$

В таблице 18 показан список эксплуатационных затрат. На рисунке 45 показано соотношение эксплуатационных затрат.

Таблица 18 – Список эксплуатационных затрат

Наименование затрат	**Затраты, руб.**
$C_{зп}$	7248
$C_{ао}$	0
$C_{то}$	28992
$C_{лс}$	0
$C_{ни}$	0
$C_{проч}$	1087
Итог	37327

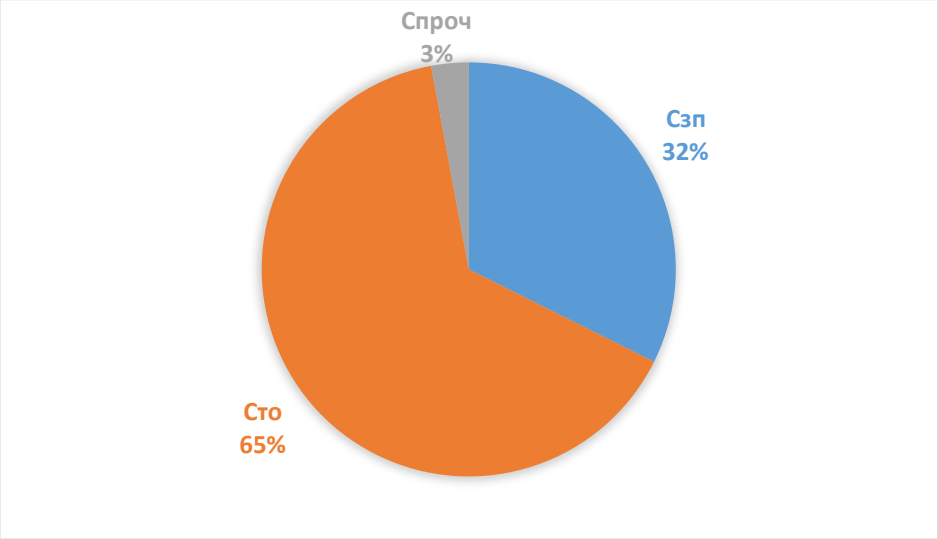

Рисунок 45 – Соотношение эксплуатационных затрат

3.4 Расчет затрат реализации проекта

Согласно методике TCO, совокупная стоимость затрат реализации проекта рассчитывается по формуле

$$TCO = DE + IC_1 + IC_2, \quad (7)$$

где DE – прямые расходы;

IC_1 – косвенные расходы первой группы,

IC_2 – косвенные расходы второй группы.

Прямые расходы рассчитываются по формуле

$$DE = DE_1 + DE_2 + DE_3 + DE_4 + DE_5 + DE_6 + DE_7 + DE_8, \quad (8)$$

где DE_1 – капитальные расходы;

DE_2 – расходы на управление информационными технологиями;

DE_3 – расходы на техническую поддержку;

DE_4 – расходы на разработку прикладного внутреннего ПО внутренними силами;

DE_5 – расходы на аутсорсинг;

DE_6 – командировочные расходы;

DE_7 – расходы на услуги связи;

DE_8 – прочие расходы;

Капитальные расходы были вычислены ранее $DE_1 = 84924$рубля;

$DE_2 = \mathrm{C_{зп}} = 7248$ рублей;

$DE_3 = \mathrm{C_{ао}} + \mathrm{C_{зп}} = 7248$ рублей;

DE_4=0, т.к. расходы на разработку прикладного ПО внутренними силами не потребуется;

DE_5 –расходы на аутсорсинг равны 0;

DE_6=0, т.к выезды в другие города не требуются.

DE_7 – связь разработчика с заказчиком происходит через Интернет или очном формате, т.к. расходы на эту связь входят в повседневные траты не связанные с разработкой проекта, то DE_7=0.

DE_8 – прочие расходы составляют 3% от суммы всех прямых расходов, т.е $DE_8 = 1721$ рубль.

По формуле 8 были получены прямые затраты:

$$DE = 84924 + 7248 + 7248 + 2983 = 102403 \text{ рубля.}$$

В таблице 19 представлен список прямых затрат.

Таблица 19 – Список прямых затрат

Наименование затрат	**Затраты, руб**
DE_1	84924
DE_2	7248
DE_3	7248
DE_4	0
DE_5	0
DE_6	0
DE_7	0
DE_8	2983
Итог	102 403

IC_1, IC_2–затраты на заполнение информационной базы и обучение персонала, внесены в капитальные затраты, других косвенных затрат нет.

В таком случае по формуле (7):

$$TCO = 84924 + 37327 = 122251 \text{ рублей.}$$

3.5 Оценка рисков разработки проекта

При разработке информационной системы были рассмотрены риски:

– Изменение в требованиях используемых инструментов для разработки системы. В силу изменений в законодательстве использование некоторого программного обеспечения может оказаться под запретом, из-за чего разработку придется переносить на другое программное обеспечение, что может существенно добавить времени к разработке проекта.

– Ошибки при обновлении данных. Разрабатываемая система должна сохранять и изменять при необходимости данные об устройствах, данные действия должны происходить своевременно, иначе получится неактуальная база данных, в результате чего отчеты могут содержать ошибки разного рода.

Таблица 20 отображает информацию о рисках, рассмотренных в рамках выпускной квалификационной работы.

Таблица 20 – Оценка рисков

№	Группы рисков	Перечень рисков	Уровень влияния риска на проект	Вероятность риска	Возможность предотвращения или снижения риска
1	Риски, связанные с выбором средств разработки	Неэффективные трудозатраты для разработки проекта	Средний	Средняя	Заранее обговорить требуемые средства разработки, при наличии требований
2	Риски, связанные с выполнением проекта	Неактуальные, неверные данные в базе	Средний	Низкий	Установление правил пользования программой в медицинских учреждениях, назначение ответственных лиц за обновление информации об объектах

Чтобы избежать рисков, связанных с невостребованностью разрабатываемой ИС, следует вовлечь сотрудников в создание системы, принять к сведению их пожелания и предложения.

Чтобы избежать рисков, связанных с ошибками при обновлении данных в базе, необходимо принять правила использования системы в медицинских учреждениях, назначить ответственных за обновление данных.

3.6 Выводы по разделу «Расчёт затрат проекта «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

В третьем разделе выпускной квалификационной работы выполнен расчет затрат реализации проекта по методу ТСО. Затраты составили 107320 рублей.

Проанализированы риски, которые могут возникнуть в процессе разработки, представлены возможные мероприятия по их решению. Среди рисков выделены неэффективные трудозатраты для разработки проекта и неактуальные, неверные данные в базе данных. Предложены мероприятия по их устранению.

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы по теме «Разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ» описана деятельность ГБУЗ РХ «РМИАЦ», которая заключается в сборе, анализе и обработке данных о деятельности медицинских организаций в республике Хакасия. Создана информационно-логическая модель базы данных, описаны таблицы в созданной базе данных.Проведен сравнительный анализ средств разработки. Выбраны средства для разработки информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

Создан программный продукт и описаны этапы его создания, а именно создание базы данных, создание отчета для вывода о наполненности информации, разработка фильтров для информационной системы.

Были вычислены капитальные затраты, эксплуатационные затраты и ТСО определены риски на этапе разработки проекта и способы их решения.

Информационная система «Паспорт цифровизации ГБУЗ РХ «РМИАЦ» представляет собой веб-приложение, куда можно записывать, хранить и выводить требующуюся информацию о медицинских организациях и их внутренней инфраструктуре.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Django: официальный сайт. – 2005. – URL:https://www.djangoproject.com/(дата обращения: 20.04.2022).
2. Flask: официальный сайт. – 2010. – URL:https://flask.palletsprojects.com/en/2.1.x/(дата обращения: 20.04.2022).
3. MySQL: официальный сайт. – 2022. – URL:https://www.mysql.com/(дата обращения: 20.04.2022).
4. PHP: официальный сайт. – 2001. – URL:https://www.php.net/(дата обращения: 20.04.2022).
5. PostgreSQL: официальный сайт. – 1996. – URL:https://www.postgresql.org/(дата обращения: 20.04.2022).
6. PyCharm : The Python IDE for Professional Developers : официальныйсайт. – 2022. – URL: https://www.jetbrains.com/pycharm/ (датаобращения: 16.03.2022).
7. Python : The official home of the Python Programming Language : официальныйсайт. – 2022. – URL : https://www.python.org/ (датаобращения: 16.03.2022).
8. SQLite: официальный сайт. – 2022. – URL:https://www.sqlite.org/index.html(дата обращения: 20.04.2022).
9. Выполнение и защита выпускной квалификационной работы по направлению 09.03.03 «Прикладная информатика»: метод. указания / сост. Е. Н. Скуратенко, В. И. Кокова, И. В. Янченко ; Сиб. федер. ун-т, ХТИ – филиал СФУ. – Абакан : ХТИ – филиал СФУ, 2017.– URL:https://khti.sfu-kras.ru/documents/Библиотека/Оформление%20научно-исследовательской%20работы/Прикладная%20информатика/Скуратенко%20ЕН%20ВКР%202017%20Met_1050.pdf (дата обращения: 05.04.2022).
10. ГБУЗ РХ «Республиканский медицинский информационно-аналитический центр»: официальный сайт. – URL: https://miac.mz19.ru/miac/ (дата обращения: 15.06.2022).

11. Единая государственная информационная система в сфере здравоохранения [сайт]. – URL: https://egisz.rosminzdrav.ru/#firstPage (дата обращения: 15.06.2022).

12. ПАСПОРТ Стратегии цифровой трансформации отрасли «Здравоохранение» до 2024 года и на плановый период до 2030 года [сайт]. – URL: https://static-0.minzdrav.gov.ru/system/attachments/attaches/000/057/382/original/Стратегия_цифровой_трансформации_отрасли_Здравоохранение.pdf?1626341177 (дата обращения: 12.06.2022).

13. Республика Хакасия. Постановления. Об утверждении уставов государственных учреждений здравоохранения Республики Хакасия: Постановление Правительства Республики Хакасия от 21 августа 2007 г. №264. – URL: http://base.garant.ru (дата обращения: 05.04.2022).

14. СТУ 7.5–07–2021 СТАНДАРТ УНИВЕРСИТЕТА : Система менеджмента качества : Общие требования к построению, изложению и оформлению документов учебной деятельности : сайт / Сибирский Федеральный Университет. – Красноярск: СФУ, 2021. – URL: https://about.sfu-kras.ru/docs/8127/pdf/808588 (дата обращения: 05.04.2022).

ПРИЛОЖЕНИЕ А

Программные коды функций для вывода отчета о заполненности данных в базе данных

```
def tsvp_info(mo1):
    sqlite_connection = sqlite3.connect('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_tvsp """
    Municipal.execute(sqlite_select_query)
    record1 = Municipal.fetchall()
    g, g1 = 0, 0
    global tvspid
    tvspid = []
    for row1 in record1:
        if row1[23] == mo1:
            g1+=1
            tvspid.append(row1[0])
            if 'Заполните' in set(row1):
                g = g + 1


    data_none = str(g)
    data_none1 = str(g1)
    return data_none, data_none1
```

Рисунок А.1 – Функция для вывода количества незаполненных строк в таблице «ТВСП»

```
def diagnostic_eqip(tvspid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_diagnosticeqip"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    i,i1 = 0,0
    for row1 in record:
        for j in tvspid:
            if row1[10]== j:
                    i1+=1
                    if 'Заполните' in set(row1):
                        i+= 1

    return i,i1
```

Рисунок А.2 – Функция для вывода количества незаполненных строк в таблице «Диагностическое оборудование»

```
def network_tvsp(tvspid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_network"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    g = 0
    global networkid
    networkid = []
    for row1 in record:
        for j in tvspid:
            if row1[0]== j and 'Заполните' in set(row1):
                g = g + 1
            if row1[0] == j:
                networkid.append(row1[1])

    return g
```

Рисунок А.3 – Функция для вывода количества незаполненных строк в таблице «Сетевая инфраструктура»

```
def network_connect(networkid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_network_connect"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    m,m1 = 0,0
    for row1 in record:
        for k in networkid:
            if row1[8]== k:
                m1+=1
                if 'Заполните' in set(row1):
                    m = m + 1
    return m,m1
```

Рисунок А.4– Функция для вывода количества незаполненных строк в таблице «Информация о доступе к интернет»

```
def security_network(networkid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_security_network"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    i,i1 = 0,0

    for row1 in record:
            for k in networkid:
                if row1[6]== k:
                    i1+=1
                    if 'Заполните' in set(row1):
                        i = i + 1
    return i,i1
```

Рисунок А.5– Функция для вывода количества незаполненных строк в таблице «Защищенная сеть передачи данных»

```
def local_network(networkid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_local_network"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    i,i1 = 0,0
    m=0
    for row1 in record:
        for j in networkid:
            if row1[8]== j:
                m+=1
                if 'Заполните' in set(row1):
                    i = i + 1
    return i,m
```

Рисунок А.6– Функция для вывода количества незаполненных строк в таблице «Локальная вычислительная сеть»

```
def arm(networkid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_arm"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    g,g1 = 0,0
    global armid
    armid = []
    for row1 in record:

            for k in networkid:
                if row1[2]== k:
                    g1+=1
                    armid.append(row1[0])
                    if 'Заполните' in set(row1):
                        g = g + 1


    return g, g1
```

Рисунок А.7– Функция для вывода количества незаполненных строк в таблице «Оборудование ARM»

```
def software(armid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_software"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    g, g1 = 0, 0
    for row1 in record:

        for k in armid:
            if row1[7] == k:
                g1+=1
                if 'Заполните' in set(row1):
                    g = g + 1
    return g,g1
```

Рисунок А.8– Функция для вывода количества незаполненных строк в таблице «Программное обеспечение»

```
def systemblock(armid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_systemblock"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    g,g1 = 0,0
    for row1 in record:
        for k in armid:
            if row1[9] == k:
                g1+=1
                if 'Заполните' in set(row1):
                    g = g + 1
    return g,g1
```

Рисунок А.9– Функция для вывода количества незаполненных строк в таблице «Системный блок»

```
def monitor(armid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_monitor"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    g, g1 = 0,0
    for row1 in record:

        for k in armid:
            if row1[5] == k:
                g1+=1
                if'Заполните' in set(row1):
                    g = g + 1
    return g,g1
```

Рисунок А.10– Функция для вывода количества незаполненных строк в таблице «Монитор»

```
def keyboard(armid):
    sqlite_connection = sqlite3.connect ('db.sqlite3')
    Municipal = sqlite_connection.cursor()
    sqlite_select_query = """ SELECT * from database_keyboard"""
    Municipal.execute(sqlite_select_query)
    record = Municipal.fetchall()
    g, g1 = 0,0
    for row1 in record:

        for k in armid:

            if row1[4] == k:
                g1+=1
                if 'Заполните' in set(row1):
                    g = g + 1
    return g,g1
```

Рисунок А.11 – Функция для вывода количества незаполненных строк в таблице «Клавиатура»

Выпускная квалификационная работа выполнена мной самостоятельно. Использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

Отпечатано в одном экземпляре.

Библиография 14 наименований.

Один экземпляр сдан на кафедру.

«____» _____ 2022 г.

_____ Сагояков Александр Дмитриевич
подпись

Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Хакасский технический институт – филиал ФГАОУ ВО
«Сибирский федеральный университет»

Кафедра прикладной информатики, математики и естественно-научных дисциплин

УТВЕРЖДАЮ
Заведующий кафедрой
_____ Е. Н. Скуратенко
подпись
«17» июня 2022 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.03 Прикладная информатика

Разработка информационной системы «Паспорт цифровизации ГБУЗ РХ «РМИАЦ»

Руководитель _____ 17.06.2022 доцент, канд. физ.-мат. наук М. А. Буреева
подпись, дата

Выпускник _____ 17.06.2022 А. Д. Сагояков
подпись, дата

Консультанты
по разделам:

Экономический _____ 17.06.22г. Е. Н. Скуратенко
подпись, дата

Нормоконтролер _____ 17.06.22 В. И. Кокова
подпись, дата

Абакан 2022